ARTICLE

Search and learning for unsupervised text generation

Lili Mou

Department of Computing Science &
Alberta Machine Intelligence Institute
(Amii), University of Alberta, Edmonton,
Canada

Correspondence
Lili Mou , Department of Computing
Science & Alberta Machine Intelligence
Institute (Amii), University of Alberta,
Edmonton, AB T6G 2E8, Canada.
Email: doublepower.mou@gmail.com

Abstract
With the advances of deep learning techniques, text generation is attracting increasing interest in the artificial intelligence (AI) community, because of its wide applications and because it is an essential component of AI. Traditional text generation systems are trained in a supervised way, requiring massive labeled parallel corpora. In this paper, I will introduce our recent work on search and learning approaches to unsupervised text generation, where a heuristic objective function estimates the quality of a candidate sentence, and discrete search algorithms generate a sentence by maximizing the search objective. A machine learning model further learns from the search results to smooth out noise and improve efficiency. Our approach is important to the industry for building minimal viable products for a new task; it also has high social impacts for saving human annotation labor and for processing low-resource languages. Link to video abstract: https://youtu.be/Xir1e9g6oIc

INTRODUCTION

Text generation is a fundamental and long-lasting problem in the fields of natural language processing (NLP) and artificial intelligence (AI), with wide applications such as conversational agents, news headline generation, and grammatical error correction. It is related to the foundation of AI, and is a key component in the Turing test (Saygin, Cicekli, and Akman 2000). Early text generation systems are mainly based on rules and templates; thus, the generated text lacks flexibility and the applications are restricted to certain narrow domains, for example, report generation (Kukich 1983) and weather forecasting (Goldberg, Driedger, and Kittredge 1994; Reiter et al. 2005).

In recent years, deep learning has achieved remarkable progress in various text generation tasks (Sutskever, Vinyals, and Le 2014; Vaswani et al. 2017). Due to the high modeling capacity, deep neural networks are able to capture the complexity of language, and generate more diverse and natural texts than early rule-based systems.

However, neural models are known to be data-hungry, usually requiring massive labeled input–output pairs, known as *parallel corpora*. For example, the widely used corpus provided by the 2014 Workshop on Statistical Machine Translation contains more than four million pairs of English–German sentences (Bojar et al. 2014). This is prohibitively expensive for deep learning models being applied to new domains, new tasks, and low-resource languages.

In this paper, I will introduce our recent progress on unsupervised text generation, which does not require parallel data for training. We formulate the text generation task as a search problem, where we define a heuristic objective function, typically involving language fluency, semantic coherency, and other task-specific constraints. Then, we perform discrete local search, for example, simulated annealing (Liu et al. 2020), to generate the output sentence by maximizing the search objective. Further, a machine learning model may learn from the search results to smooth out the search noise and improve inference

efficiency. In this way, our approach is unsupervised, yet achieving high performance in a variety of tasks. Empirically, our approach outperforms previous unsupervised text generation methods to a large extent, and has a comparable performance to supervised methods for text simplification (Kumar et al. 2020). Our study largely bridges the gap between supervised and unsupervised text generation.

The rest of this paper is organized as follows: I will first discuss the social impacts of our work, followed by detailed but easy-to-understand technical explanation with examples and results. Then, I will mention a few important future directions that extend our current work, and finally conclude this paper.

SOCIAL GOOD

Unsupervised text generation can benefit our society in various aspects.

First, unsupervised approaches are important to the AI industry, which typically requires that a minimal viable product is seen before investing substantial resources, including funding and human labor. Therefore, our unsupervised text generation techniques are particularly suitable for new tasks and startup companies, where massive annotated data are either unavailable or not affordable.

Second, unsupervised text generation will help low-resource language processing. Training supervised deep learning NLP models requires massive labeled corpora, which only exist for several most-spoken languages. This prohibits the applications of the deep learning technique to low-resource languages. Our approaches, on the contrary, do not require parallel data, and can be potentially applicable to various text generation tasks in different languages.

Last but not least, our unsupervised techniques save human labor for data annotation. In machine learning, massive labeled data are often annotated by mechanical Turks, and this could be expensive and time-consuming; asking annotators to write sentences for text generation tasks is especially cumbersome and often yields poor-quality corpora. Our unsupervised text generation largely reduces human labor, as we do not require parallel data. Even if human-written text is needed for certain tasks, our approach is able to provide an initial draft for mechanical Turks to edit.

METHODOLOGY

We tackle unsupervised text generation by search approaches, where we heuristically define a scoring function that roughly estimates the quality of a candidate output sentence given some input in a certain task. Then, we perform discrete local search to maximize the score for unsupervised text generation. This is accomplished by iteratively proposing a local edit of the candidate sentence, such as word insertion, deletion, and replacement. The proposed candidate may either be accepted or rejected depending on its score, although a better sentence is more likely to be accepted. In this way, we can gradually search for a high-scored sentence as the output text.

Search objectives

In a standard supervised machine learning application, the task is defined by data sets. For example, if the training corpus contains pairs of sentences of the same meaning but different wordings, then the trained machine learning model will accomplish the paraphrase generation task. However, defining a task by data sets is not feasible in the unsupervised setting, as we do not have parallel corpora. We observe that, in many text generation tasks, the quality of an output text can often be decomposed into several aspects, each of which can be modeled relatively easily. Thus, we may define the task by a heuristically designed scoring function, as explained below.

We would like the generated text to be fluent, so we use a language model to evaluate the language fluency of a given candidate sentence $\mathbf{y}$, denoted by $s_{\text{fluency}}(\mathbf{y})$. A language model is trained to maximize the likelihood of a training corpus (Jurafsky and Martin 2009); the underlying assumption here is that if a sentence is more fluent, then its likelihood is higher, and vice versa. The language model can be trained either on a task-specific unlabeled corpus using a recurrent neural network (Mikolov et al. 2010) or on large-scale generic corpora with the Transformer architecture (Radford et al. 2019). Usually, fine-tuning a pretrained language model on a task-specific corpus will yield the highest performance. It is noted that none of the above variants require labeled parallel corpora.

Then, we consider the semantic of the generated sentence given some input. In various text generation tasks, the semantic of the output should be close to the input; examples of such applications include paraphrase generation and summarization. We leverage the *embedding* technique (Mikolov et al. 2013), which essentially maps an object (e.g., a word, a phrase, or a sentence) into a vector space. The cosine of two embedding vectors indicates how close two objects are. Again, the embeddings can be given by either recurrent neural networks (Pagliardini, Gupta, and Jaggi 2018) or by pretrained language models (Liu et al. 2019). Moreover, we find that in certain applications, we may enhance the semantic measure by the embeddings of keywords (Li et al. 2020) and entities (Kumar et al. 2020) to better preserve the content. In general, we denote the

semantic scorer by $s_{\text{semantic}}(\mathbf{y}, \mathbf{x})$ for a candidate output $\mathbf{y}$ given input $\mathbf{x}$.

There may be additional task-specific constraints, and we denote our third scorer by $s_{\text{task}}(\mathbf{y})$. In fact, the task-specific constraint can be either a soft or a hard constraint, varying largely based on the nature of the task at hand. For keywords-to-sentence generation, $s_{\text{task}}(\mathbf{y})$ is simply an indicator function, representing whether $\mathbf{y}$ contains the keywords or not (Miao et al. 2019); for text summarization, $s_{\text{task}}(\mathbf{y})$ indicates whether the output is within the length budget or not (Schumann et al. 2020); and for paraphrasing, $s_{\text{task}}(\mathbf{y})$ is counting the fraction of the overlap between input and output sentences (Papineni et al. 2002), as a paraphrase cannot be the same as the input.

We may also utilize external engineering for the task-specific constraint. The text simplification task appears similar to summarization, but emphasizes the simplicity of the output, instead of the length. In our study (Kumar et al. 2020), we adopt the Flesch reading ease score (Kincaid et al. 1975), which involves manually designed features, such as the sentence length and the number of syllables per word. Another interesting task is style-transfer generation, which aims to change the style of a sentence, while keeping the content, for example, changing an informal sentence to a formal one (Rao and Tetreault 2018). In Li et al. (2020), we train a classifier in a supervised way based on style labels, serving as the task-specific scorer $s_{\text{task}}(\mathbf{y})$. It is noted that the style label is much easier to obtain than parallel corpora of sentence pairs, and our generation still works in an unsupervised manner.

In summary, our overall scoring function consists of language fluency, semantic similarity, and other task-specific constraints, having the following form:

$$s(\mathbf{y}, \mathbf{x}) = s_{\text{fluency}}(\mathbf{y}) \cdot s_{\text{semantic}}(\mathbf{y}, \mathbf{x}) \cdot s_{\text{task}}(\mathbf{y})$$

Here, we consider the multiplications of all the individual scorers because their scales may be different. If a scorer can be negative, it should be normalized to guarantee positivity. Generally speaking, our scorer estimates the fitness of a candidate sentence given an input for some task. It serves as our search objective for unsupervised text generation.

Search algorithms

We search towards the optimum of the objective function to obtain an output sentence. In supervised text generation, it is typical to apply beam search (Sutskever, Vinyals, and Le 2014), which searches for a word at a time but keeps a fix-sized beam to prevent the search space growing exponentially. This works well with the supervised machine learning model, where the joint probability of a sentence is decomposed into word-level probabilities, thus providing step-by-step guidance for the beam search. By contrast, our objective function yields a score only based on a complete sentence, instead of a partial sentence. Thus, traditional beam search is inappropriate in our scenario.

We observe, on the other hand, that the output and input resemble each other in various text generation applications. For example, a paraphrase is close to the original sentence, except that a few words and phrases are changed to synonyms and that the syntactic structure may be modified slightly; a summary even overlaps with the input text largely, although clutters and unimportant information are dropped. Thus, we propose to tackle unsupervised text generation by local search. In other words, we start from the input sentence and iteratively perform local editing to the candidate sentence to maximize the objective scoring function.

A local search algorithm is a two-stage process of proposal and acceptance. First, an edit operation is proposed from a candidate operation set, and then the algorithm will either accept or reject the proposal depending on the score of the resulting sentence.

Our general operations of local editing include word replacement, insertion, and deletion. At each search step, we randomly select one of the edit operations. For deletion, we simply remove the selected word. For insertion and replacement, we need to propose a candidate word from the vocabulary. We propose to sample a word according to the Gibbs distribution (Koller and Friedman 2009) that is induced from our scorer, that is to say, a word that yields a better sentence should be more likely to be sampled. Such a proposal is more efficient than uniformly proposing a word from the vocabulary.

The editing operations may also be designed specifically to the task. In our study on text simplification (Kumar et al. 2020), we identify two major types of editing for simplifying a sentence: substituting a common word and re-ordering phrases. We use the WordNet (Miller 1995), a lexical database, to obtain the synonyms, and substitute a rare word with a more frequently appearing synonym. For phrase re-ordering, we parse the sentence to a constituency parse tree (Manning et al. 2014), which is a hierarchical organization of sentence components (e.g., noun phrases, verb phrases, and clauses). We manipulate the nodes of the parse tree for phrase re-ordering. It is noted that the proposed edit need not be perfect, as a corrupted sentence will be filtered out in the acceptance phrase.

Another simple yet interesting example of edit operation is for text summarization (Schumann et al. 2020). We notice that, in our application, a summary is oftentimes a subset of the input, even with the word order preserved. Therefore, we generate a summary by word-level extraction. Suppose we have a budget of k words for
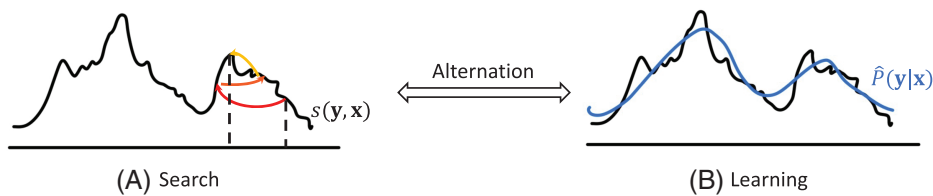
FIGURE 1 Overview of our search-and-learning approach to unsupervised text generation. Alternations may be performed between search and learning to boost performance. The diagram is adapted from Li et al. (2020)

the summary, our system determines which k words to be extracted from the input as the summary. This largely reduces the output space and simplifies the search process. Our edit operation is simply swapping the selection and nonselection of two words.

Regarding the acceptance stage, our first study (Miao et al. 2019) adopts the Metropolis–Hastings algorithm (Metropolis et al. 1953; Hastings 1970). This is essentially a sampling procedure that yields unbiased samples from the score-induced distribution. Although it is tempting to sample a sentence from a properly defined distribution for applications like paraphrasing, we later realize that optimization (rather than sampling) will lead to higher empirical performance due to better convergence; thus, we adopt search algorithms in our later studies.

Our mostly commonly used search algorithm is simulated annealing (Van Laarhoven and Aarts 1987), which also works in a propose-and-accept manner (Liu et al. 2020; Li et al. 2020; Dong et al. 2021). Generally, if the proposed sentence has a higher score, then the sentence is directly accepted. If the proposed sentence has a lower score, then it tends to be rejected. However, a worse sentence may still be accepted to better explore the search space, controlled by a temperature hyperparameter. If the temperature is high, then the algorithm is more exploratory, accepting more low-scored sentences. If the temperature is low, the algorithm is more greedy. Simulated annealing starts from a high temperature but gradually cools down during the search process, analogous to chemical annealing. This differs from Metropolis–Hastings sampling, because simulated annealing aims to converge to an optimum.

In certain applications, a hill-climbing algorithm may also work well. Hill climbing is greedy search that only accepts better sentences. We adopt hill climbing for summarization, where we have design a simpler search space (Schumann et al. 2020), and for simplification, where we have dedicated engineering on edit operations (Kumar et al. 2020).

Figure 1A illustrates the scoring function $s(\mathbf{y}, \mathbf{x})$ and our search algorithm. Table 1 further summarizes the design of search objectives, edit operations, and search algorithms for different tasks in our studies.

Learning from search results

Admittedly, search-based text generation has its own disadvantages. First, our search objective is heuristically defined by several decomposed evaluation criteria. Although it correlates with output quality in the population level, the objective function may not be suitable for every single data sample. Moreover, our algorithm is slow when deployed because the search process requires several hundred iterations of proposals and re-evaluations. This prevents our approach from real-time applications.

To address the above drawbacks, we propose in Li et al. (2020) to train a machine learning model that learns from the search results. Specifically, we adopt a pretrained language model (Radford et al. 2019) and fine-tune it based on the input and the search algorithm's output to estimate the probability $\hat{P}(\mathbf{y}|\mathbf{x})$.

As known, a machine learning model pools together the knowledge of the individual samples, and thus is able to smooth out the noise of our heuristically defined objective function, demonstrated in Figure 1B. Empirically, our trained machine learning model immediately achieves higher performance than the search algorithm. In addition, this gives us 5–10 times speed-up in the deployment because the machine learning model predicts an output sentence in a word-by-word fashion.

Since our machine learning model outputs a better sentence than search, we may further feed it as the initial search candidate. It serves as a more meaningful starting point for the search algorithm than copying the input sentence. We alternate the search and learning processes to boost the performance.

Our idea of search and learning is similar to reinforcement learning (Sutton and Barto 2018). For example, the AlphaGo system performs Monte Carlo tree search for the Go game, and trains a neural network model from search results, outperforming world-class professional human Go players (Silver et al. 2017, 2018). We instead perform

TABLE 1 Applications of unsupervised text generation

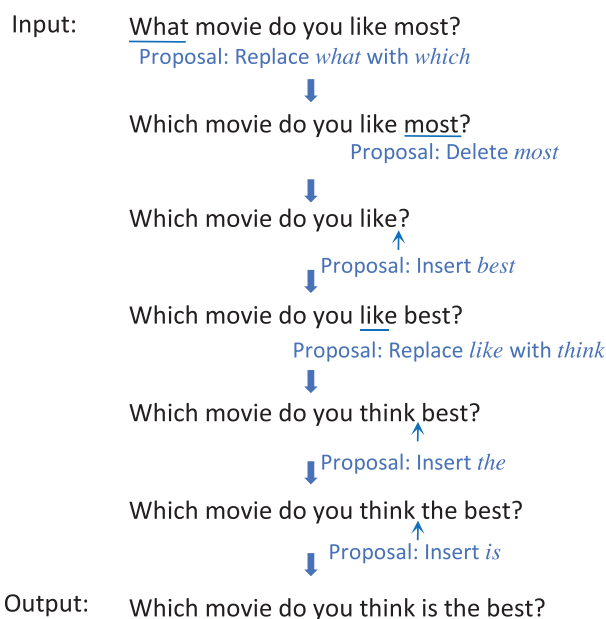| Task	Task-specific constraint	Edit operation	Algorithm	Study
Grammatical error correction	None	Word editing	Metropolis–Hastings sampling	Miao et al. (2019)
Paraphrasing	Different wordings	Word editing	Simulated annealing	Liu et al. (2020)
Summarization	Length	Word selection	Hill climbing	Schumann et al. (2020)
Simplification	Ease score (Kincaid et al. 1975)	Word/syntactical editing	Hill climbing	Kumar et al. (2020)
Style-transfer generation	Style accuracy	Word editing	Simulated annealing	Li et al. (2020)

TABLE 2 Examples of our unsupervised text generation in selected applications. References are the provided output sentences for the evaluation purpose; they are not used for training in our unsupervised setting.

Task	Keywords-to-sentence generation (Miao et al. 2019)
Input	Lottery, scholarships
Output	But the lottery has provided scholarships.
Task	Grammatical error correction (Miao et al. 2019)
Input	Even if we are failed, we have to try to get a new things.
Reference	Even if we all failed, we have to try to get new things.
Output	Even if we are failing, we have to try to get some new things.
Task	Paraphrasing (Liu et al. 2020)
Input	Where are best places for spring snowboarding in the US?
Reference	Where is the best place to snowboard in the US?
Output	Where can I find the best places in the US for snowboarding?
Task	Summarization (Schumann et al. 2020)
Input	A German registered container ship ran aground at the entrance to the French port of Le Havre early Tuesday, but authorities said there were no casualties.
Reference	Container ship runs aground in French port.
Output	A container ship ran aground but there were no casualties.

local search, such as simulated annealing, based on the characteristics of our own problems.

Applications and results

Our search-and-learning approach is a generic framework for unsupervised text generation, and can be applied to a variety of applications. Table 2 shows sample output

Input: What movie do you like most?
Proposal: Replace *what* with *which*

Which movie do you like most?
Proposal: Delete *most*

Which movie do you like?
Proposal: Insert *best*

Which movie do you like best?
Proposal: Replace *like* with *think*

Which movie do you think best?
Proposal: Insert *the*

Which movie do you think the best?
Proposal: Insert *is*

Output: Which movie do you think is the best?

FIGURE 2 An example of the editing process for paraphrase generation (Miao et al. 2019). The rejected edits are omitted for clarity

in four tasks, namely, keywords-to-sentence generation, grammatical error correction, paraphrase generation, and summarization. For example, the sentence "*Where are best places for spring snowboarding in the US?*" is paraphrased as "*Where can I find the best places in the US for snowboarding?*" where our approach performs nontrivial editing. As seen, our approaches indeed generate meaningful output in each task, although it may be different from the reference output.

Figure 2 further demonstrates the dynamics of our editing in the paraphrase generation task. Given an input sentence "*What movie do you like most?*," our approach suggests meaningful edits for paraphrasing, such as replacing *what* with *which* and replacing *most* with *best*. Although the intermediate sentences may not be perfect (e.g., "*Which movie do you think best?*"), our approach is able to further insert the words *is the* to make the sentence

TABLE 3 Performance of paraphrase generation on a Quora dataset. The iBLEU metric measures the similarity against the reference, penalized by the similarity against the input; a higher score indicates a better model. Our search and learning results are quoted from Li et al. (2020).

Category	Method	iBLEU
Supervised methods	Qian et al. (2019)	14.83
	Fine-tuning GPT-2 (Radford et al. 2019)	19.19
Distant supervision	Round-trip translation	14.36
	Domain adaptation (Li et al. 2019)	10.36
Unsupervised methods	Variational autoencoder (Bowman et al. 2016)	8.16
	Our search approach	14.52
	Our search and learning	17.48

fluent. The best candidate along the search process will be selected as the output.

Quantitatively speaking, our search and learning approaches achieve high performance in all the above applications. Table 3 lists the performance of our approaches compared with various competing models in different categories. As seen, both search-based and search-and-learning systems largely outperform the previous unsupervised paraphrase generator, given by a variational autoencoder (Bowman et al. 2016). Although our unsupervised performance is worse than fine-tuning the pretrained language model (Radford et al. 2019), we outperform a recent competitive supervised system based on reinforcement learning (Qian et al. 2019). In text simplification, we see a similar trend that our unsupervised approach achieves comparable performance to supervised systems (Kumar et al. 2020).

In the industry, round-trip translation is widely used for paraphrasing, and in turn, for data augmentation. The idea is to make use of a machine translation system, translating a sentence into a foreign language and translating it back to obtain a paraphrase. This requires parallel data of translation rather than paraphrasing, known as *distant supervision*. Our search-and-learning approach also largely outperforms the paraphrasing system based on round-trip translation. I fully believe that our search-and-learning framework will become a widely adopted industrial practice in the near future, as it largely bridges the gap between supervised and unsupervised text generation.

FUTURE DIRECTIONS

The work described in this paper presents a novel idea in the NLP field. It also points to several important future directions of both fundamental and applied research.

One fundamental research question is how to perform efficient search in the sentence space, which can be inspired by the reinforcement learning community. Our framework resembles the search and learning in reinforcement learning (Sutton and Barto 2018). However, text generation has its own challenges compared with other domains, such as videos games and the Go game. First, we have a larger branching factor than even the very challenging Go game; that is to say, for each generation step, we may have tens of thousands of words to consider from the entire vocabulary. Second, reinforcement learning highly depends on the reward function (i.e., the scoring function in our paper), which serves as the entire training signal. For example, the reward in a game could be the binary indicator of success or failure; in a typical reinforcement learning application, the reward function is often accurate, albeit noisy. However, our scoring function may be deterministically incorrect for certain samples, as it is defined in a heuristic way. It remains unclear how these factors will affect the search process.

Sha (2020) proposes a gradient-based approach for searching in the lexical space. The gradient provides informative guidance for the proposal. However, Sha's current approach requires a differentiable scoring function, and it only works in the word level. It would be a promising direction to generalize the gradient-based editing in future work.

Another important extension is to propose edit operations beyond the word level. As seen in Figure 2, local editing yields low-quality intermediate candidate sentences because our search algorithm may not be greedy, and this helps to explore a wide range of the sentence space. However, such editing proposals are more likely to be rejected due to the calculation of the acceptance rate. Therefore, it is tempting to propose editing in the phrase level, for example, replacing the phrase *like most* to *think is the best* in one step of editing. Although we have attempted syntax-based phrasal editing in Kumar et al. (2020), our current treatment is still rudimentary as we only support phrase re-ordering. Future research may propose phrasal insertion and replacement by generating a plausible candidate phrase given a certain position in a sentence.

Currently, our approach is also restricted in that most of our tasks require the input and output closely resembling each other. This may be relaxed with additional heuristics based on the task. In unsupervised machine translation,

for example, we may obtain an initial candidate translation by dictionary lookup of every word. Then, our approach can be applied to edit the candidate translation.

For certain tasks, a completely unsupervised approach may not be feasible because it is hard to decompose a scoring function into several aspects by heuristics. For example, it is difficult to articulate what constitutes a high-quality response in a dialog system. Nevertheless, we may apply our search and learning approach to address other research questions. In a recent study of ours (Dong et al. 2021), we adopt the search algorithm to control the emotion of a response. Traditional methods feed the desired emotion as input information, which may be ignored by the generation system (Zhou et al. 2018). By contrast, we directly edit a candidate response to ensure the presence of the desired emotion, thus generating much richer and more emotional responses than other dialog systems. In Jolly, Dengel, and Mou (2021), we address few-shot data-to-text generation, where the goal is to generate a description based on tabular data with a few samples. We observe that, in the few-shot setting, important information is often missing in the output sentence. Thus, we adopt the search-and-learning approach, where we insert the missing information to improve semantic coverage, and further learn from the search results to improve fluency and efficiency. In this way, we only use 1% of parallel data but achieve comparable performance to supervised learning with the entire data set.

Our work also opens wide applications beyond the natural language domain. In a recent study (Liu et al. 2021), we extend the simulated annealing to graph processing in the domain of molecule optimization. Our goal is to improve the hydrophobicity of a molecule, and we manage to design a scoring function that estimates hydrophobicity and molecule similarity; we further propose edit operations on graphs including node and edge insertion, deletion, and replacement. Experimental results show that, compared with previous molecule optimization methods, our approach significantly improves the hydrophobicity of a molecule by a series of local edits, while preserving other chemical characteristics. This further shows the generality of our approach for different types of data and in different domains. The key to the applications is the design of search objective and edit operations.

CONCLUDING REMARKS

In this paper, I present our recent progress on search and learning approaches to unsupervised text generation, where we do not have parallel data for training. We define a heuristic scoring function that estimates the quality of the output text given input for a certain task. Then, we formulate text generation as a search problem, and adopt discrete search to maximize the objective score. We may further train a machine learning model from the search results to smooth out search noise and improve inference efficiency, while keeping the unsupervised nature.

We have applied our framework to a variety of applications, including paraphrase generation, summarization, text simplification, keywords-to-sentence generation, grammatical error correction, and style-transfer generation. Our approach can be further adapted to other controllable text generation in the supervised or few-shot settings. I also point out a few important future directions, including efficient search in the sentence space and applications to other domains beyond natural language.

Our search and learning approaches to unsupervised text generation will largely benefit the AI industry, because the unsupervised nature is helpful in building new products, especially for small businesses. Our work will also have positive social impacts in processing low-resource languages and saving human labors for data annotation.

CONFLICT OF INTEREST
The author declares that there is no conflict.

ORCID
Lili Mou https://orcid.org/0000-0001-7753-4295

REFERENCES
Bojar, O., C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, et al. 2014. "Findings of the 2014 workshop on statistical machine translation." In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, 12–58.

Bowman, S. R., L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. 2016. "Generating sentences from a continuous space." In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 10–21.

Dong, C., C. Huang, O. Zaïane, and L. Mou. 2021. "Simulated annealing for emotional dialogue systems." In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 2984–8.

Goldberg, E., N. Driedger, and R. Kittredge. 1994. "Using natural-language processing to produce weather forecasts." *IEEE Expert* 9(2): 45–53.

Hastings, W. K. 1970. "Monte Carlo sampling methods using Markov chains and their applications." *Biometrika* 57: 97–109.

Jolly, S., A. Dengel, and L. Mou. 2021. "Search and learn: improving semantic coverage for data-to-text generation." In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*, 10858–66.

Jurafsky, D., and J. H. Martin. 2009. *Speech and Language Processing*, 2nd ed. Harlow: Pearson.

Kincaid, J. P., R. P. Fishburne Jr, R. L. Rogers, and B. S. Chissom. 1975. "Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel." Technical report, Naval Technical Training Command, Research Branch, Millington TN.

Koller, D., and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. Cambridge: MIT Press.

Kukich, K. 1983. "Design of a knowledge-based report generator." In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, 145–50.

Kumar, D., L. Mou, L. Golab, and O. Vechtomova. 2020. "Iterative edit-based unsupervised sentence simplification." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7918–28.

Li, J., Z. Li, L. Mou, X. Jiang, M. Lyu, and I. King. 2020. "Unsupervised text generation by learning from search." In *Advances in Neural Information Processing Systems*, 10820–31.

Li, Z., X. Jiang, L. Shang, and Q. Liu. 2019. "Decomposable neural paraphrase generation." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3403–14.

Liu, X., P. Li, F. Meng, H. Zhou, H. Zhong, J. Zhou, L. Mou, and S. Song. 2021. "Simulated annealing for optimization of graphs and sequences." *Neurocomputing* 465: 310–24.

Liu, X., L. Mou, F. Meng, H. Zhou, J. Zhou, and S. Song. 2020. "Unsupervised paraphrasing by simulated annealing." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 302–12.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. "RoBERTa: a robustly optimized BERT pretraining approach." *arXiv preprint arXiv:1907.11692*.

Manning, C. D., M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. 2014. "The Stanford CoreNLP natural language processing toolkit." In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 55–60.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. 1953. "Equation of state calculations by fast computing machines." *The Journal of Chemical Physics* 21(6): 1087–1092.

Miao, N., H. Zhou, L. Mou, R. Yan, and L. Li. 2019. "CGMH: constrained sentence generation by Metropolis-Hastings sampling." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 6834–42.

Mikolov, T., M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. 2010. "Recurrent neural network based language model." In *Proceedings of Interspeech*, 1045–8.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. "Distributed representations of words and phrases and their compositionality." In *Advances in Neural Information Processing Systems*, 3111–9.

A Miller, G. 1995. "WordNet: a lexical database for english." *Communications of the ACM* 38(11): 39–41.

Pagliardini, M., P. Gupta, and M. Jaggi. 2018. "Unsupervised learning of sentence embeddings using compositional n-gram features." In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 528–40.

Papineni, K., S. Roukos, T. Ward, and W.-J. Zhu. 2002. "Bleu: a method for automatic evaluation of machine translation." In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–8.

Qian, L., L. Qiu, W. Zhang, X. Jiang, and Y. Yu. 2019. "Exploring diverse expressions for paraphrase generation." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 3173–82.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. "Language Models are Unsupervised Multitask Learners." OpenAI (blog).

Rao, S., and J. Tetreault. 2018. "Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer." In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 129–40.

Reiter, E., S. Sripada, J. Hunter, J. Yu, and I. Davy. 2005. "Choosing words in computer-generated weather forecasts." *Artificial Intelligence* 167: 137–69.

Saygin, A. P., I. Cicekli, and V. Akman. 2000. "Turing test: 50 years later." *Minds and Machines* 10(4): 463–518.

Schumann, R., L. Mou, Y. Lu, O. Vechtomova, and K. Markert. 2020. "Discrete optimization for unsupervised sentence summarization with word-level extraction." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5032–42.

Sha, L. 2020. "Gradient-guided unsupervised lexically constrained text generation." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 8692–703.

Silver, D., T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, and et al. 2018. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." *Science* 362(6419): 1140–4.

Silver, D., J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, and et al. 2017. "Mastering the game of Go without human knowledge." *Nature* 550(7676): 354–9.

Sutskever, I., O. Vinyals, and Q. V. Le. 2014. "Sequence to sequence learning with neural networks." In *Advances in Neural Information Processing Systems*, 3104–12.

Sutton, R. S., and A. G. Barto. 2018. *Reinforcement Learning: An Introduction*. Cambridge: MIT Press.

Van Laarhoven, P. J., and E. H. Aarts. 1987. *Simulated Annealing: Theory and Applications*. Springer. Dordrecht, Netherland.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. "Attention is all you need." In *Advances in Neural Information Processing Systems*, 5998–6008.

Zhou, H., M. Huang, T. Zhang, X. Zhu, and B. Liu. 2018. "Emotional chatting machine: emotional conversation generation with internal and external memory." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 730–8.

AUTHOR BIOGRAPHY

Lili Mou is an Assistant Professor at the Department of Computing Science, University of Alberta. He is also an Alberta Machine Intelligence Institute (Amii) Fellow and a Canada CIFAR AI (CCAI) Chair. Lili received his BS and PhD degrees in 2012 and 2017, respectively, from School of EECS, Peking University. After that, he worked as a Postdoctoral Fellow at the University of Waterloo. His research interest is mainly in machine learning methods for natural language processing. He has more than 50 publications at top conferences and journals, such as AAAI, ACL, EMNLP, ICML, ICLR, NeurIPS, and TACL. He has also presented conference tutorials at EMNLP-IJCNLP'19 and ACL'20. He received a New Faculty Highlight Award in the AAAI'21 conference.