Evaluating Curriculum Learning Strategies in Neural Combinatorial Optimization

Michal Lisicki
University of Guelph
Vector Institute for AI
mlisicki@uoguelph.ca

Arash Afkanpour
Google
arashaf@google.com

Graham W. Taylor
University of Guelph
Vector Institute for AI
gwtaylor@uoguelph.ca

Abstract

Neural combinatorial optimization (NCO) aims at designing problem-independent and efficient neural network-based strategies for solving combinatorial problems. The field recently experienced growth by successfully adapting architectures originally designed for machine translation. Even though the results are promising, a large gap still exists between NCO models and classic deterministic solvers, both in terms of accuracy and efficiency. One of the drawbacks of current approaches is the inefficiency of training on multiple problem sizes. Curriculum learning strategies have been shown helpful in increasing performance in the multi-task setting. In this work, we focus on designing a curriculum learning-based training procedure that can help existing architectures achieve competitive performance on a large range of problem sizes simultaneously. We provide a systematic investigation of several training procedures and use the insights gained to motivate application of a psychologically-inspired approach to improve upon the classic curriculum method.

1 Introduction

Attention and sequence-to-sequence models have recently been shown capable of approximating solutions to combinatorial problems [3, 19, 28]. However, a large performance gap still exists between those models and the exact [1] or heuristic [5, 12, 17, 22, 24] solvers. So far, researchers have mostly focused on training separate models for each problem size. The ability of a model to perform well across a larger range of sizes simultaneously has been mostly overlooked.

Training and testing on individual problem sizes misleadingly diminishes the performance gap between neural combinatorial optimization (NCO) solutions and solvers. While algorithms are often designed to provide bounded approximation [5, 17, 24], NCO requires an ensemble of models to achieve similar performance, which can be prohibitive in industrial applications [2].

Curriculum learning (CL) is based on the premise that neural networks learn faster from tasks gradually increasing in complexity [4, 9]. CL algorithms used for scheduling tasks are called *sampling strategies*. The naïve strategy often leads to catastrophic forgetting and decline of efficiency. Adaptive staircase [21], a strategy developed in psychophysics, incorporates rehearsal and adjustment of difficulty to the current capacity, which were shown to significantly improve performance in multi-task learning [20]. In this work we assess CL as an effective training procedure to help a recent state-of-the-art NCO model [19] achieve competitive performance on a range of problem sizes simultaneously.

We provide a novel application of the CL framework in NCO. Our assessment of the model's performance on problem sizes other than it was trained on shows that the extent of knowledge that can be transferred between tasks changes smoothly with problem size. Investigation of several baseline strategies shows a clear benefit from rehearsal and training on problems of increasing difficulty. These insights serve to motivate a psychologically-inspired approach that improves upon classic CL.

1st Workshop on Learning Meets Combinatorial Algorithms @ NeurIPS 2020, Vancouver, Canada.

2 Background

Travelling Salesperson Problem. The travelling salesperson problem (TSP) is a combinatorial problem in which we look for the shortest path through n cities, visiting each city exactly once, and return to the origin. The NCO model we utilize was designed to operate on the Euclidean TSP, which requires the distances between the cities to be defined using the Euclidean metric. Imposing such a restriction allows for clear visualizations and direct comparison of the results with previous research. The problem was shown to be NP-complete [1], but in practice we can find solutions within a reasonable time frame, using highly optimized solvers, like Concorde [1].

TSP can be formulated as a search for an optimal permutation of input cities. Following the notation from [19], let $\boldsymbol{\pi} = \{\pi_1, \dots, \pi_n\}$, where each element $\pi_i \in \{1, \dots, n\}$ is a node index, such that $\pi_i \neq \pi_j \ \forall i \neq j$. Given a distance matrix D, where input elements D_{ij} correspond to a cost of travel between a city i and a city j, our objective is to find a permutation resulting in a tour of minimal length $L(\boldsymbol{\pi}) = \sum_{i=1}^{n} D_{\pi_i \pi_{i+1}}$.[1]

Evaluation Metric. Comparing the performance of combinatorial problems that differ in complexity is challenging and often requires domain-specific knowledge. A commonly used metric is the approximation ratio $R(x, y) = \frac{f(y)}{\text{OPT}(x)}$ between the score $f(y)$ of an approximate feasible solution y to a problem instance x, and the score of an optimal solution $\text{OPT}(x)$ [16, 18]. Constant factor approximation algorithms are expected to be no worse than a given ratio, independent of the problem size. For TSP the objective $f(y) = L(\boldsymbol{\pi})$. Most of the previous work in NCO [3, 19, 28] uses a variant of the approximation ratio, called the *optimality gap* $\big(R(x, y) - 1\big)$, and propose using a Concorde solution as a reference point in the ratio [3, 19, 28]. In this work we calculate the optimality gap w.r.t. the Held-Karp (HK) lower bound [11], as it is independent of any software, well understood in the combinatorial optimization community [6], and easy to implement.

Neural Combinatorial Optimization. Most of the recent advances in NCO are based on sequence-to-sequence models [3, 15, 28], attention models [19], or graph neural networks[2] (GNN) [8, 14]. Solutions inspired by successful applications in machine translation [25] do not carry forward directly to TSP, as the the models depend on a fixed dictionary. Pointer Networks [28] train an attention mechanism to point back to the input, which eschews the need for a separate dictionary and allows for training on graphs of variable size. Reinforcement learning was later shown to provide a more suitable specification of the objective function for combinatorial problems than supervised learning [3, 19]. The attention model (AM), proposed by [19], builds upon the empirical evidence that pure attention-based models tend to outperform their recurrent neural network counterparts on sequence prediction tasks [7, 26], mainly due to improvements in algorithmic efficiency and regularization effects. While graph convolutional networks show better performance on individual problem sizes, the attention-based methods provide comparable results with better generalization [13, 14].

Curriculum Learning. Curriculum learning (CL) is a training procedure that utilizes knowledge transfer between tasks of increasing difficulty and devises an efficient task sampling strategy. While the difficulty of a problem can be defined with respect to the capability of a model to solve it, we impose a further restriction and assume a global ordering of tasks, with a direct correspondence between the problem's size and its difficulty level. CL attemps to exploit a neural network's ability to facilitate learning solutions to difficult tasks with the knowledge of simpler ones. Recently, there has been a lot of focus on algorithms that can automate the process of designing curricula for specific applications, particularly those stemming from reinforcement learning [10, 23].

Adaptive staircase. Adaptive staircase is an improved CL strategy adapted from psychophysics to deep reinforcement learning [21]. At each step, the strategy determines if it is either: ready to advance to a more difficult task, backtrack, or train on a task at the current difficulty level. In the base case, the agent is always trained on a task of the current difficulty level t. The alternative is a probe case, where the task is sampled uniformly from a range $[1, t]$. The base and probe cases are chosen equiprobably at each trial. The level is incremented to $t + 1$ when the agent achieves a certain performance threshold α after observing N_{base} base trials. Probe trials contribute to training, but the agent is not evaluated on them. The number of base trials per task can either be constant or can be proportional to the current difficulty level: $N_{\text{base}} \propto t$.

[1]The additional element $\pi_{n+1} = \pi_1$ in the equation ensures the closure of the path.

[2]The authors of [19] point out that their model is equivalent to the graph attention network proposed in [27].

3 Experiments and discussion

To test the effectiveness of CL in improving NCO performance on a range of problem sizes, we performed two experiments. We started by analyzing the ability of the AM model [19] to solve tasks on which it was not trained. The results showed the limitations of knowledge transfer between the models, which prompted our investigation of several different sampling strategies: (i) single task, (ii) stochastically sampled range of tasks, and (iii) two curriculum learning approaches. In all the experiments we used REINFORCE with a greedy rollout baseline with a learning rate $\eta = 10^{-3}$. Training each of the methods for 100 epochs took ≈ 50 hours on 4 P100 GPUs, depending of the dominant size of the problem. Following [19], we initialized parameters with Uniform$(-1/\sqrt{d}, 1/\sqrt{d})$ for the input dimension d. Every epoch the network used 2,500 batches of 512 samples.[3] As a metric we chose the optimality gap, reported as percentage, between the average cost returned by the model and the HK lower bound estimate computed over 10^4 TSPs per problem size. For the smallest problem sizes (4–9) we ran exhaustive search over 10^5 TSPs to reduce the variability.

To quantify the capability of the model to generalize to other tasks we trained 147 models from TSP 4 to 150, and tested them on sizes 10 to 300. HK lower bounds for this experiment were extrapolated from the available representative set. Each of the models was initialized with the parameters of a model pretrained in a classic curriculum fashion, and subsequently trained for 20 epochs. Fig. 2 shows the performance matrix, in a form of a heat map, where each row corresponds to the performance of a model trained on a single task when tested on a range of tasks from 10 to 300.

In the second experiment, we compared six sampling strategies (Fig. 1). Following previous work [3, 19, 28], we first trained three models on fixed problem sizes 20, 50 and 100. For the rest of the strategies, we trained the model on a set of TSPs from 4 (the minimal meaningful problem size) to 100. We started with a baseline (uniform) stochastic model that selected a task randomly at each epoch. Then we tested the two CL approaches — the classic approach of monotonically increasing problem sizes, and the adaptive staircase.

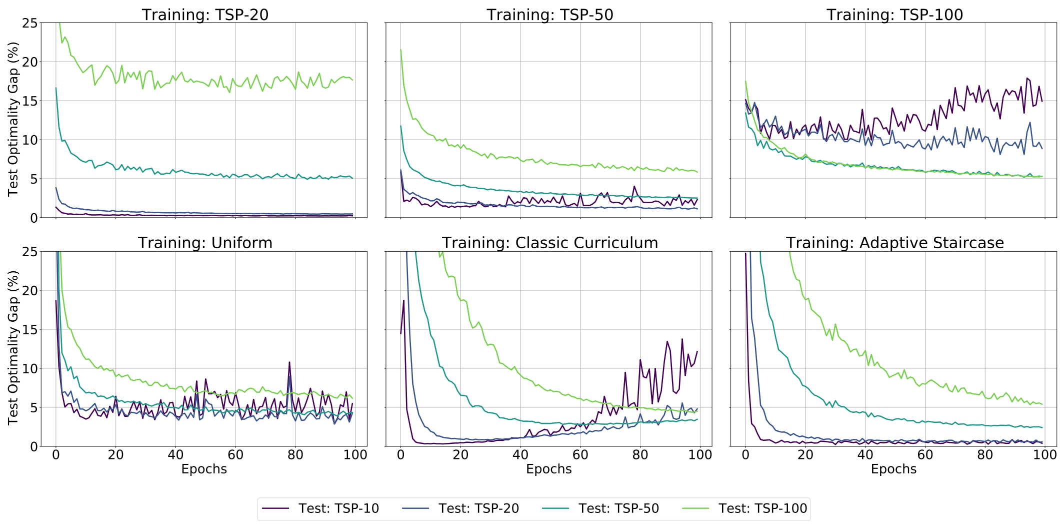

Figure 1: For each sampling strategy, we follow the training procedure of the Attention Model (AM) proposed in [19]. The curves are color coded from dark to bright in increasing level of task difficulty. Each subplot title denotes the task schedule on which the model was trained. The "Test: TSP-N" curve represents validation of the model on a task N at every epoch.

We start the discussion by observing that the performance of models trained on neighboring tasks is correlated (Fig. 3, Fig. 2). The figures show that the performance of different models can be delineated by a skewed Gaussian centered at the task on which the model was trained. The ability of the model to solve problems close in size suggests a large knowledge transfer, which occurs with widening spread but smaller magnitude as the problem size increases (Fig. 2). Moreover, a drift from smaller problem sizes implies that a successful sampling strategy must be able to balance small and large problem sizes.

[3]In case of adaptive staircase the number varied per epoch. This is explained later in the section.

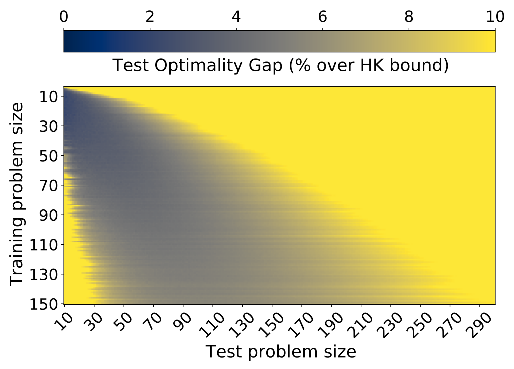

Figure 2: Heat map of validation scores of models trained separately on problem sizes 4 to 150. The optimality gap is measured with respect to the HK lower bound and the threshold caps it from the top at 10%.

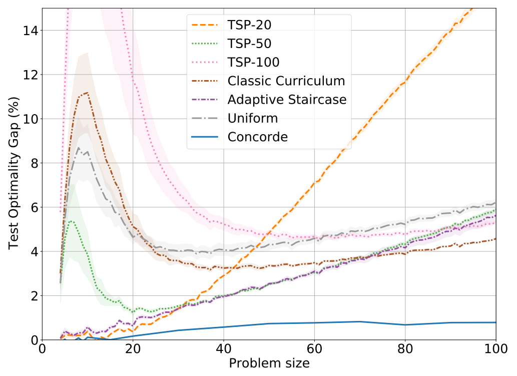

Figure 3: The optimality gap of the final epoch performance of each of the models w.r.t. exhaustive search (sizes 4–9) and HK bound (sizes 10–150). The confidence intervals represent 2 standard errors over 5 runs.

Throughout the course of training (Fig. 1), the models that do not visit the previous problems along with the new ones show initial improvement in performance, but as soon as the network reaches a certain epoch the performance decays for the tasks that are further away. This shows the limitation of knowledge transfer and exposes catastrophic forgetting.

The model trained with uniform sampling (Fig. 3) is the closest in performance to a constant factor approximator, but its optimality gap across all the problem sizes is larger than most of the other methods. We conclude that balancing the problem sizes naïvely does not lead to an overall improvement in performance. Knowledge transfer discrepancies, catastrophic forgetting and lack of efficiency prompted our investigation of curriculum-based methods.

Classic curriculum learning consolidates knowledge from observed tasks and delivers a more consistent performance across all the tasks as compared to the baseline. However, the method still results in catastrophic forgetting (Fig. 1), as the cues learned from solving smaller problem sizes are not helpful enough to the model to retain that knowledge. One way to alleviate forgetting is to combine rehearsing with incremental learning.

We propose to use adaptive staircase, a method previously applied to train a reinforcement learning agent in a psychophysics environment [21]. The experiment was run by scheduling tasks according to the probe case with $\alpha = 0.05$ threshold on the optimality gap to advance the task difficulty. If performance α was not attained after a full epoch we decremented the task. We set the number of base trials (batches) per epoch to *(10+current_graph_size - min_graph_size) / (max_graph_size - min_graph_size)*. To make sure that the number of probe trials did not exceed the number of batches available to the other sampling strategies, we stopped training one epoch earlier than the other strategies. This resulted in $\approx 1.6\%$ reduction in the amount of data exposed to this strategy. Fig. 1 and Fig. 3 both show that adaptive staircase results in the best trade-off between forgetting and average performance across problem sizes.

4 Conclusion

In this work we have assessed the ability of an attention-based NCO model to perform well on a large range of problem sizes. We found that knowledge transfer between tasks is measurable and changes smoothly, but is otherwise limited and cannot be ignored when designing a sampling strategy. Curriculum learning provides a convenient framework that takes into account complexity and knowledge transfer between tasks. The classic approach of training on tasks monotonically increasing in complexity suffers from catastrophic forgetting. Our analysis demonstrates the need for rehearsal of less complex samples during training. By choosing an efficient rehearsal-based sampling strategy, adaptive staircase, we were able to show both the usefulness of curriculum learning in neural combinatorial optimization and provide important evidence supporting the applicability of a biologically-inspired method to an emerging area of deep learning.

Acknowledgments and Disclosure of Funding

The authors would like to thank Vithursan Thangarasa for preliminary discussions. We also thank Irwan Bello from Google Brain as well as our lab colleagues, Magdalena Sobol and Angus Galloway, for their support in reviewing this manuscript.

References

[1] David L. Applegate, Robert E. Bixby, Vasek Chvatal, and William J. Cook. Concorde TSP solver, 2006. URL `www.math.uwaterloo.ca/tsp/concorde`.

[2] David L. Applegate, Robert E. Bixby, Vasek Chvatal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006. ISBN 9780691129938. URL `http://www.jstor.org/stable/j.ctt7s8xg`.

[3] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *International Conference on Learning Representations*, 2017. URL `https://arxiv.org/abs/1611.09940`.

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[5] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[6] Nicos Christofides. Combinatorial optimization. In *A Wiley-Interscience Publication, Based on a series of lectures, given at the Summer School in Combinatorial Optimization, held in Sogesta, Italy, May 30th-June 11th, 1977, Chichester: Wiley, 1979, edited by Christofides, Nicos*, 1979.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`.

[8] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

[9] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[10] Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1311–1320. PMLR, 2017. URL `http://proceedings.mlr.press/v70/graves17a.html`.

[11] Michael Held and Richard M Karp. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6):1138–1162, 1970. URL `https://www.cse.wustl.edu/~ychen/7102/Karp-TSP.pdf`.

[12] Keld Helsgaun. An extension of the lin-kernighan-helsgaun TSP solver for constrained traveling salesman and vehicle routing problems. Technical report, Technical Report, Roskilde University, 2017. URL `https://forskning.ruc.dk/en/publications/an-extension-of-the-lin-kernighan-helsgaun-tsp-solver-for-constra`.

[13] Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint arXiv:1906.01227*, 2019.

[14] Chaitanya K Joshi, Quentin Cappart, Louis-Martin Rousseau, Thomas Laurent, and Xavier Bresson. Learning TSP requires rethinking generalization. *arXiv preprint arXiv:2006.07054*, 2020.

[15] Yoav Kaempfer and Lior Wolf. Learning the multiple traveling salesmen problem with permutation invariant pooling networks. *arXiv preprint arXiv:1803.09621v2*, 2019. URL `https://arxiv.org/abs/1803.09621v2`.

[16] Viggo Kann. *On the approximability of NP-complete optimization problems*. PhD thesis, Royal Institute of Technology Stockholm, 1992.

[17] Anna R Karlin, Nathan Klein, and Shayan Oveis Gharan. A (slightly) improved approximation algorithm for metric TSP. *arXiv preprint arXiv:2007.01409*, 2020.

[18] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems*, pages 6348–6358, 2017. URL `https://arxiv.org/abs/1704.01665`.

[19] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019. URL `https://arxiv.org/abs/1803.08475`.

[20] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks, 2019. URL `https://arxiv.org/abs/1909.08383`.

[21] Joel Z Leibo, Cyprien de Masson d'Autume, Daniel Zoran, David Amos, Charles Beattie, Keith Anderson, Antonio García Castañeda, Manuel Sanchez, Simon Green, Audrunas Gruslys, et al. Psychlab: a psychology laboratory for deep reinforcement learning agents. *arXiv preprint arXiv:1801.08116*, 2018.

[22] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973. URL `https://www.jstor.org/stable/pdf/169020.pdf`.

[23] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep rl: A short survey. *arXiv preprint arXiv:2003.04664*, 2020.

[24] A. I. Serdjukov. Some extremal bypasses in graphs [in Russian]. *Upravlyaemye Sistemy*, 17: 76–79, 1978.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. URL `https://arxiv.org/abs/1409.3215`.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. URL `https://arxiv.org/abs/1706.03762`.

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL `https://arxiv.org/abs/1710.10903`.

[28] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015. URL `https://arxiv.org/abs/1506.03134`.