Neural Combinatorial Optimization with Heavy Decoder: Toward Large Scale Generalization

25.01.16

김지훈

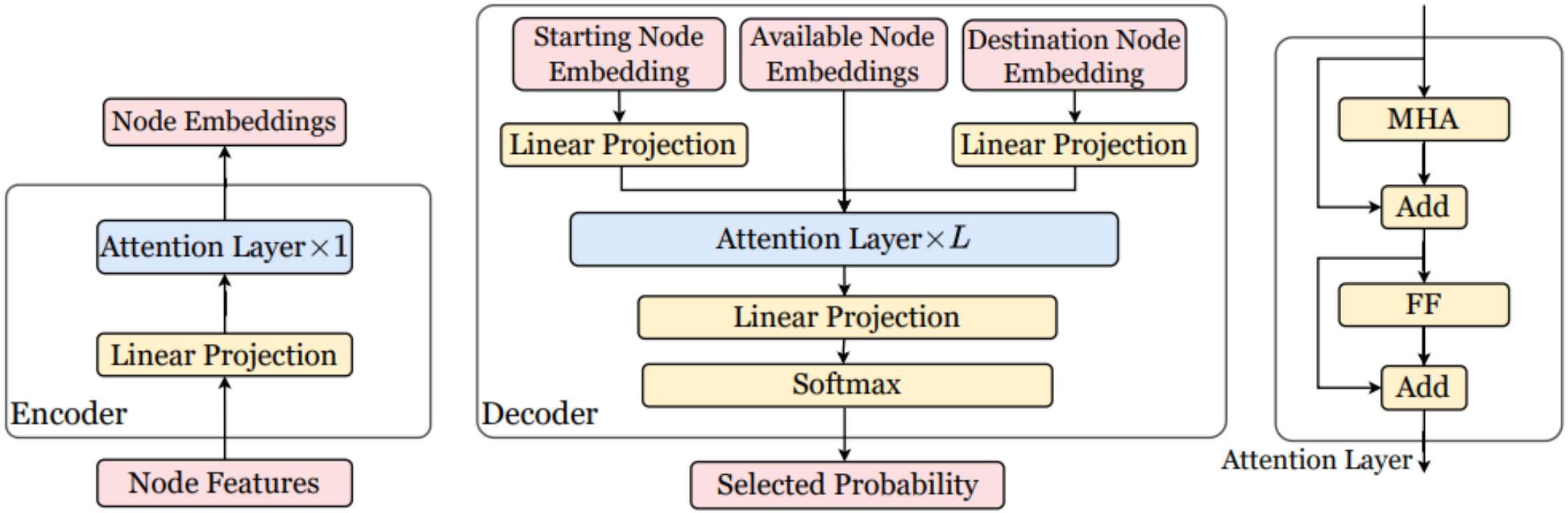
I Introduction

❖ 대규모 조합 최적화를 위한 새로운 신경망 모델

- 대규모 인스턴스 크기를 가진 조합최적화 (Combinatorial Optimization; CO) 문제는 기존의 신경망 문제로 해결할 수 없음
- 본 연구에서는 대규모 인스턴스를 해결하기 위해 강력한 일반화 성능을 가진 새로운 신경망 모델인 경량 인코더 및 헤비 디코더 (Light Encoder and Heavy Decoder; LEHD) 모델을 제안

Light Encoder and Heavy Decoder (LEHD)

❖ 전체 모델 구조



Ⅱ Light Encoder and Heavy Decoder (LEHD)

❖ Heavy Decoder

- 본 논문은 기존 AM (Kool et al.) 기반 모델들이 정적 노드 임베딩 행렬만 가지고 있기 때문에 디코딩 과정 내내 캡처하는 노드 간의 관계가 업데이트되지 않아 문제 크기가 커질 수록 관계성을 나타내는 능력이 악화된다고 설명

AM

LEHD

Light Encoder and Heavy Decoder (LEHD)

❖ Heavy Decoder

- $\boldsymbol{W_1}$ 과 $\boldsymbol{W_2}$ 그리고 $\boldsymbol{W_O}$ 는 각각 학습 가능한 파라메터
  - ✓ $\boldsymbol{W_1}$: 출발지 노드를 구분하기 위한 레이어
  - ✓ $\boldsymbol{W_2}$: 가장 직전에 선택된 노드를 구분하기 위한 레이어
  - ✓ $\boldsymbol{W_O}$: 선택 확률을 계산하기 위해 임베딩 행렬을 변환하는 벡터

$$
\begin{aligned}
\widetilde{H}^{(0)} &= \mathrm{Concat}(W_1\mathbf{h}_{x_1}^{(1)}, W_2\mathbf{h}_{x_{t-1}}^{(1)}, H_a), \\
\widetilde{H}^{(1)} &= \mathrm{AttentionLayer}(\widetilde{H}^{(0)}), \\
&\cdots \\
\widetilde{H}^{(L)} &= \mathrm{AttentionLayer}(\widetilde{H}^{(L-1)}), \\
u_i &= \begin{cases} W_O\widetilde{\mathbf{h}}_i^{(L)}, & i \neq 1 \text{ or } 2 \\ -\infty, & \text{otherwise} \end{cases}, \\
\mathbf{p}^t &= \mathrm{softmax}(\mathbf{u}),
\end{aligned}
$$

Supervised Learning with Data Augmentation

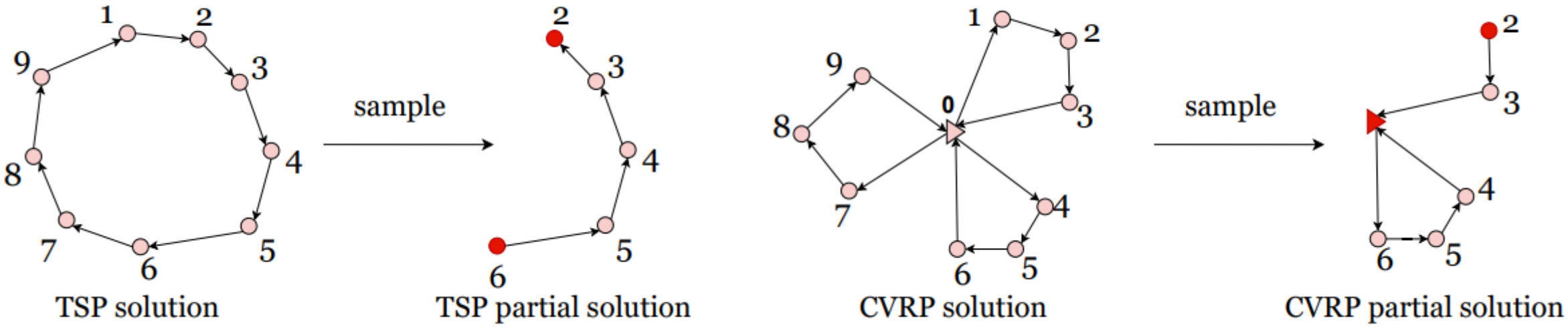
❖ Heavy Decoder의 단점과 지도학습

- 헤비 디코더는 인코더 만큼의 데이터들이 디코더에 반복적으로 할당되기 때문에 기존 모델에 비해 연산량이 급증함.
- 시퀀스가 완성되어야 가중치의 업데이트가 일어나는 AM의 강화학습 (Reinforcement Learning; RL)은 대규모 인스턴스에 적합하지 않음
- 논문에서는 이러한 문제를 해결하기 위해 강화학습 대신 지도학습 (Supervised Learning; SL)과 데이터 증강(Data Augmentation; DA)을 사용하여 해결함

Supervised Learning with Data Augmentation

❖ 부분 최적해

- 최적해는 그 일부들도 최적임.
- 이를 이용하여 하나의 최적해로부터 다양한 크기의 부분해를 샘플링 할 수 있음 (실험에서는 최소 노드 4개 이상부터 부분해를 생성).
- 샘플링 된 데이터를 SL의 정답 데이터로 활용
- 데이터들의 크기가 다양하기 때문에 이 또한 일반화에 이점으로 작용



Supervised Learning with Data Augmentation

❖ Training

- SL로 학습하는 LEHD는 입력이 다음과 같이 변경됨

LEHD with RL　　LEHD with SL

- 미분 기울기를 계산하기 위한 *loss* 계산이 달라짐

$$-(advantage)\sum_{i=1}^{u}\log(p_i)$$

LEHD with RL

$$-\sum_{i=1}^{u} y_i \log(p_i)$$

LEHD with SL

Supervised Learning with Data Augmentation

❖ Inference

- 입력 방식

Step 1 LEHD with SL

- 선택되지 않은 노드들이 없어질 때까지 반복
- 랜덤으로 선택되는 초기 노드들을 샘플링 기법을 통해 반복적으로 재구성한 뒤 재구성된 해의 품질이 좋으면 기존 솔루션을 대체하는 RRC (Random Re-Construct) 메커니즘 제안

Experiment

❖ 실험 구성

- 최적해는 Concorde Solver와 HGS를 통해서 도출
- 학습과 추론은 모두 NVIDIA GeForce RTX 3090 GPU with 24GB memory. 사용
- 레이어 수, 헤드 수, 임베딩 차원 및 피드포워드 네트워크의 차원 수 등은 AM 모델과 동일
- 도시 100개의 TSP 및 CVRP 인스턴스 100만개를 학습 1024 배치 크기로 학습

❖ TSP

		TSP100		TSP200		TSP500		TSP1000	
Concorde		0.000%	34m	0.000%	3m	0.000%	32m	0.000%	7.8h
LKH		0.000%	56m	0.000%	4m	0.000%	32m	0.000%	8.2h
OR-Tools		2.368%	11h	3.618%	17m	4.682%	50m	4.885%	10h
Att-GCN+MCTS*		0.037%	15m	0.884%	2m	2.536%	6m	3.223%	13m
MDAM bs50		0.388%	21m	1.996%	3m	10.065%	11m	20.375%	44m
POMO augx8		0.134%	1m	1.533%	5s	22.187%	1m	40.570%	8m
SGBS		0.060%	40m	0.562%	4m	11.550%	54m	26.035%	7.4h
EAS		0.057%	6h	0.496%	28m	17.08%	7.8h	-	-
BQ greedy		0.579%	0.6m	0.895%	3s	1.834%	0.4m	3.965%	2.4m
BQ bs16		0.046%	11m	0.224%	1m	0.896%	6m	2.605%	38m
LEHD greedy		0.577%	0.4m	0.859%	3s	1.560%	0.3m	3.168%	1.6m
LEHD RRC	50	0.0284%	7.4m	0.123%	0.6m	0.482%	3.4m	1.416%	22m
	100	0.0114%	13.7m	0.0761%	1.2m	0.343%	8m	1.218%	43m
	300	0.0044%	40m	0.0363%	3.3m	0.223%	22m	0.899%	2.1h
	500	0.0025%	1.1h	0.0280%	5.3m	0.193%	37m	0.818%	3.5h
	1000	**0.0016%**	2.2h	**0.0182%**	10.5m	**0.167%**	1.2h	**0.719%**	7h

Experiment

❖ CVRP

		CVRP100		CVRP200		CVRP500		CVRP1000	
LKH3		0.000%	12h	0.000%	2.1h	0.000%	5.5h	0.000%	7.1h
HGS		-0.533%	4.5h	-1.126%	1.4h	-1.794%	4h	-2.162%	5.3h
OR-Tools		6.193%	2h	6.894%	1h	9.112%	2.2h	11.662%	3h
MDAM bs50		2.211%	25m	4.304%	3m	10.498%	12m	27.814%	47m
POMO augx8		0.689%	1m	4.866%	7s	19.901%	1m	128.885%	10m
SGBS		0.079%	40m	2.581%	1m	15.343%	16m	136.980%	2.3h
EAS		**-0.234%**	15h	0.640%	33m	11.042%	9.3h	-	-
BQ greedy		2.993%	0.7m	3.527%	4s	5.121%	0.4m	9.812%	2.4m
BQ bs16		0.611%	10m	1.141%	0.6m	2.991%	6m	7.784%	39m
LEHD greedy		3.648%	0.5m	3.312%	3s	3.178%	0.3m	4.912%	1.6m
LEHD RRC	50	0.535%	7.2m	0.515%	0.6m	0.930%	8m	2.814%	27m
	100	0.272%	17m	0.217%	1.1m	0.546%	14m	2.370%	45m
	300	0.029%	52m	-0.146%	3.6m	0.045%	36m	1.582%	2.3h
	500	-0.044%	1.4h	-0.246%	6m	-0.107%	1h	1.270%	4h
	1000	-0.112%	2.8h	**-0.383%**	11.3m	**-0.347%**	2h	**0.921%**	8h

❖ Benchmark instances

			POMO	BQ		LEHD	
	Size	#	aug×8	greedy	bs16	greedy	RRC
TSPLib	<100	6	0.792%	1.076%	0.505%	0.976%	**0.481%**
	100-200	21	2.423%	2.684%	1.318%	2.336%	**0.158%**
	200-500	15	13.413%	3.177%	2.183%	2.742%	**0.200%**
	500-1k	6	31.678%	8.311%	5.521%	4.049%	**1.310%**
	>1k	22	63.810%	42.566%	36.730%	11.267%	**4.088%**
	All	70	26.439%	15.668%	12.923%	5.260%	**1.529%**
			POMO	BQ		LEHD	
	Set (size)	#	aug×8	greedy	bs16	greedy	RRC
CVRPLib	A (31-79)	27	4.970%	6.310%	1.627%	5.871%	**0.647%**
	B (30-77)	23	4.747%	6.859%	2.221%	6.049%	**0.812%**
	E (12-100)	11	11.402%	5.884%	1.211%	4.809%	**0.541%**
	F (44-134)	3	15.973%	12.568%	7.404%	9.051%	**3.009%**
	M (100-199)	5	4.861%	8.407%	3.691%	7.094%	**1.817%**
	P (15-100)	23	15.525%	5.902%	2.393%	6.611%	**0.917%**
	X (100-1k)	100	21.684%	12.526%	9.774%	12.520%	**3.511%**
	All	192	15.450%	9.692%	6.153%	9.465%	**2.253%**

❖ RL vs SL

	TSP50	TSP100	TSP200	TSP500	TSP1000
RL	5.372%	7.891%	12.581%	25.377%	46.441%
SL	**0.375%**	**0.789%**	**1.545%**	**3.500%**	**6.566%**

	(10k instances) TSP100		TSP200		(1k instances) TSP500		TSP1000	
Concorde	0.000%	34m	0.000%	23m	0.000%	4h	0.000%	61h
POMO augx8	0.134%	1m	1.459%	0.6m	21.948%	8m	40.551%	1.1h
POMO augx8-sample100	0.080%	1.7h	1.609%	1.2h	37.285%	16h	82.015%	117h
POMO augx8-RRC100	0.115%	2m	1.283%	1.6m	11.900%	11m	25.378%	1.2h
POMO augx8-RRC1000	0.075%	12m	0.820%	8m	6.753%	29m	16.261%	2.1h
LEHD greedy	0.577%	0.4m	0.849%	0.2m	1.585%	2.1m	3.089%	13m
LEHD RRC 100	0.0114%	13.7m	0.053%	6.5m	0.357%	58m	1.195%	5.3h
1000	**0.0016%**	2.2h	**0.015%**	1h	**0.179%**	9.5h	**0.735%**	57h