2025 ICML International Conference On Machine Learning Workshop on MAS

Agentic Predictor

Performance Prediction for Agentic Workflows via Multi-View Encoding

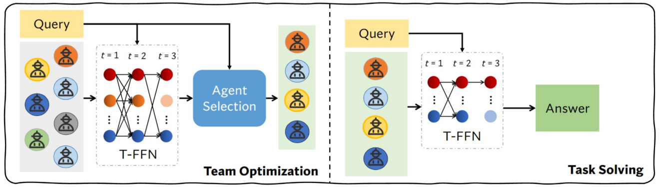
Patara Trirat[1] Wonyong Jeong[1] Sung Ju Hwang[1 2]

[1]DeepAuto.ai [2]KAIST

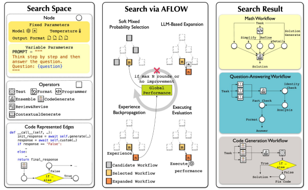
https://deepauto-ai.github.io/agentic-predictor

Why Predict Agentic Workflow Performance?

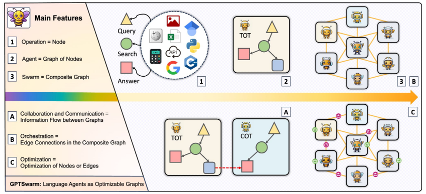
- Multi-agent systems powered by LLMs show great promise.



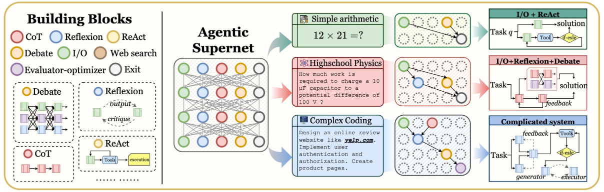
DyLAN
[COLM'24]



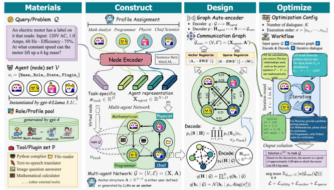
AFlow
[ICLR'25]



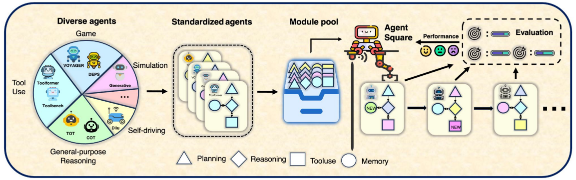
GPTSwarm
[ICML'24]



MaAS
[ICML'25]



G-Designer
[arXiv'24]



AgentSquare
[ICLR'25]

Why Predict Agentic Workflow Performance?

- Current optimization frameworks rely on costly execution-based (runtime or LLM calls) evaluations.

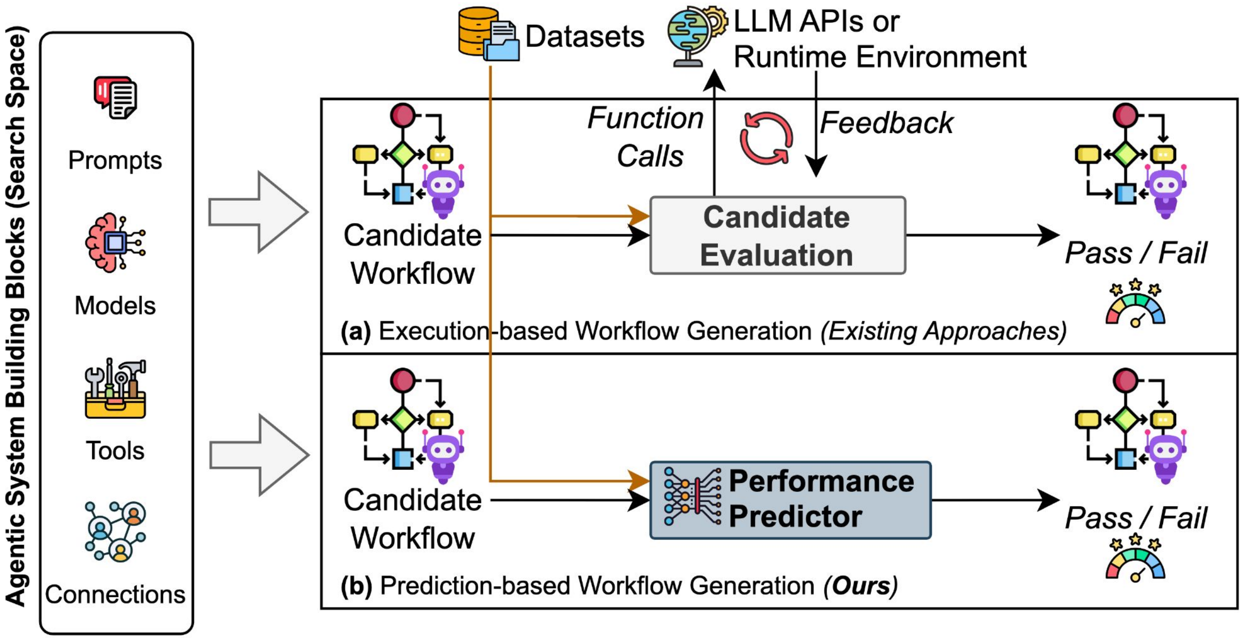
Table 3. Efficiency comparison between MaAS and state-of-the-art baselines on the MATH Benchmark. We shade the values of the lowest token/cost/wall-clock time and the highest performance.

Method	Training				Inference				Overall
	Prompt token	Completion token	Total cost ($)	Wall-clock time (min)	Prompt token	Completion token	Total cost ($)	Wall-clock time (min)	Acc. (%)
LLM-Debate	-	-	-	-	3, 275, 764	10, 459, 097	6.76$	92	48.54
DyLAN	22, 152, 407	16, 147, 052	13.01$	508	6, 081, 483	3, 303, 522	2.89$	39	48.63
MacNet	-	-	-	-	7, 522, 057	2, 043, 600	2.35$	47	45.18
GPTSwarm	21, 325, 266	6, 369, 884	7.02$	129	3, 105, 571	788, 273	0.93$	30	47.88
AFlow	33, 831, 239	29, 051, 840	22.50$	184	2, 505, 944	2, 151, 931	1.66$	23	51.28
MaAS	3, 052, 159	2, 380, 505	3.38$	53	1, 311, 669	853, 116	0.42$	19	51.82

Table extracted from MaAS [ICML'25]

Why Predict Agentic Workflow Performance?

- A predictive model can estimate the quality and viability of agentic workflows.



Challenges in Agentic Workflow Prediction

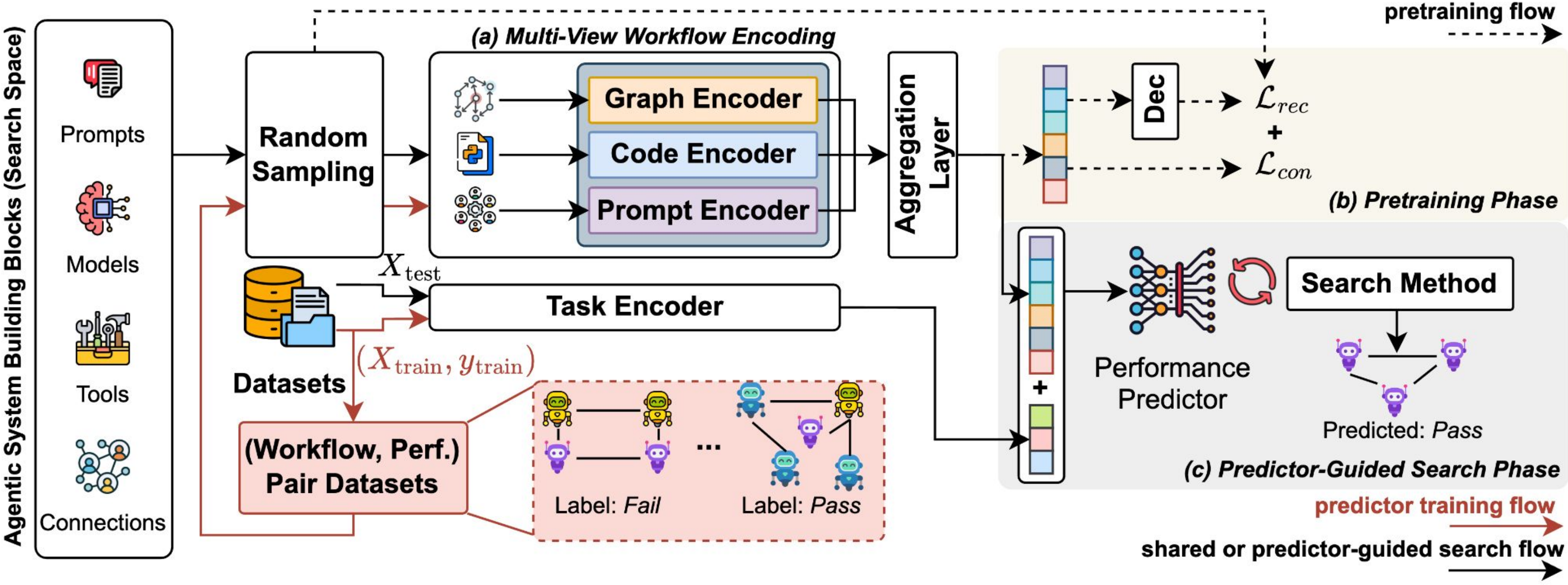
- **Heterogeneity**
  - Workflows vary widely in communication, code, and prompts.
- **Scarcity of Labeled Data**
  - Only few labels available due to expensive evaluations.

Our Solution: *Agentic Predictor*

- **Multi-View Encoding** integrates graph, code, and prompt views.
- **Cross-Domain Unsupervised Pretraining** uses unlabeled data from diverse tasks.

- **Lightweight Predictor** quickly estimates workflow performance, guiding efficient search.

Architecture Overview



Multi-View Workflow Encoding

- **Graph View** explicitly captures structural dependencies and direct interactions among agents, emphasizing interagent communication channels.
- **Code View** implicitly encodes complex semantic structures, logical sequences, computational complexities, and patterns of tool usage inherent in workflow implementations.
- **Prompt View** provides semantic embeddings that encapsulate nuanced agent roles, behavioral descriptions, and broader contextual guidance embedded within system and instruction prompts.

Multi-View Workflow Encoding: *Encoder Network*

aggregates importance scores across different views
captures inter-graph importance

- (Multi-)Graph View

$$\mathbf{Z}_{\mathcal{G}} = \mathrm{AttenPool}(\mathrm{CrossGraphAttn}([\mathrm{GNN}(\mathcal{G}_{\text{prompt}}),\ \mathrm{GNN}(\mathcal{G}_{\text{code}}), \mathrm{GNN}(\mathcal{G}_{\text{operator}})])),$$

- Code View

$$\mathbf{Z}_{\mathcal{C}} = \mathrm{MLP}_{\mathcal{C}}(\mathcal{C})$$

- Prompt View

$$\mathbf{Z}_{\mathcal{P}} = \mathrm{MLP}_{\mathcal{P}}(\mathcal{P})$$

- Aggregation Layer

$$\mathbf{Z} = \mathrm{MLP}([\mathbf{Z}_{\mathcal{G}}, \mathbf{Z}_{\mathcal{C}}, \mathbf{Z}_{\mathcal{P}}])$$

Cross-Domain Unsupervised Pretraining

- In real-world scenarios, the availability of labeled performance data for agentic workflows may be highly limited due to the costly and time-consuming evaluation process.
- To address this challenge and enable data-efficient predictor training, we *optionally* adopt a two-phase strategy.
- **Cross-Domain Multi-Task Pretraining**

$$\mathcal{L}_{rec} = \frac{1}{M}\sum_{i=1}^{M} \|\mathcal{G}_i - \hat{\mathcal{G}}_i\|^2 + \|\mathcal{C}_i - \hat{\mathcal{C}}_i\|^2 + \|\mathcal{P}_i - \hat{\mathcal{P}}_i\|^2$$

+

$$\mathcal{L}_{con} = \frac{1}{M}\sum_{i=1}^{M} -\log \frac{\exp(\mathrm{sim}(\mathbf{Z}_i, \mathbf{Z}_j^{+})/\tau)}{\sum_{k=1}^{M}\exp(\mathrm{sim}(\mathbf{Z}_i, \mathbf{Z}_k)/\tau)}$$

Positive pairs are drawn from configurations that solve the same task successfully, while *negatives* include task-mismatched or failed configurations.

This learning objective encourages the encoder to capture meaningful signals related to both structure and performance.

Performance Predictor

- **Input Features**
  - The **learned workflow embeddings** from the previous stage.
  - The **task-specific semantic embeddings** from natural language task descriptions. These embeddings, derived from pretrained language models (e.g., T5 or BERT).
- **Loss Function:** binary cross-entropy loss for *fail* vs. *pass* predictions

$$\mathcal{L}_{\text{pred}} = -\frac{1}{N}\sum_{i=1}^{N}\left[e_i \log \hat{e}_i + (1 - e_i)\log(1 - \hat{e}_i)\right]$$

Experiments

- **(Q1)** How does Agentic Predictor perform as a predictor of agentic workflow performance compared to relevant baselines?
- **(Q2)** Is the pretraining phase helpful for maintaining prediction quality under varying numbers of labels?
- **(Q3)** How do different design choices and configurations of Agentic Predictor affect its predictive accuracy?

Experiments: *Setup*

- **Benchmark Dataset**: FLORA-Bench (Zhang et al., 2025b), including ***five*** representative datasets across ***three*** core domains in the agentic workflow literature:
  - code generation (HumanEval, MBPP),
  - mathematics problem solving (GSM8K, MATH), and
  - reasoning (MMLU).

Table 2. Summary of benchmark statistics.

Domains	Code Generation	Math Problem	Reasoning Task
# workflows	38	42	30
Average # nodes	6.11	5.49	6.58
# tasks	233	782	2400
# samples	7362	4059	72000

Experiments: *Setup*

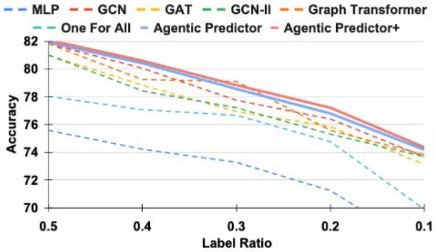
- **Evaluation Metrics**:
  - **Accuracy** quantifies how well a model predicts agentic workflow performance.
  - **Utility** evaluates the *consistency between the workflow rankings predicted by the model and the ground-truth rankings*, emphasizing the model's ability to determine the relative order of different workflows.

Experiments: ***Results*** *(Prediction Accuracy (Q1))*

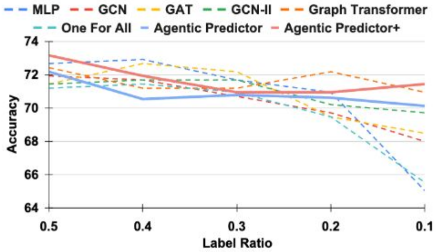
Table 3. Performance comparison between Agentic Predictor and baseline methods. The best and second-best results are highlighted in **bold** and <u>underlined</u>, respectively.

Domain	Code Generation		Math Problem		Reasoning Task		Average	
Model	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility
MLP	78.02±0.59	73.94±1.35	73.73±0.31	<u>69.64±0.29</u>	78.45±0.08	88.48±0.63	76.73±0.33	77.35±0.76
GCN	84.35±0.34	72.73±3.18	76.19±0.42	66.52±1.66	87.12±0.14	**91.82±0.46**	82.55±0.30	77.02±1.77
GAT	84.49±0.56	76.46±0.91	<u>76.44±0.61</u>	66.51±1.28	87.07±0.08	89.40±0.68	<u>82.67±0.42</u>	<u>77.46±0.96</u>
GCN-II	83.72±0.40	<u>77.75±1.98</u>	75.04±0.31	64.33±0.47	<u>87.28±0.14</u>	89.92±1.90	82.01±0.28	77.33±1.45
Graph Transformer	<u>84.71±0.45</u>	74.09±0.35	75.45±0.23	66.48±0.96	86.93±0.27	90.60±1.97	82.36±0.32	77.06±1.09
One For All	81.05±0.34	73.42±1.39	75.21±0.23	69.08±0.64	82.52±0.13	87.64±1.98	79.59±0.23	76.71±1.34
Agentic Predictor	**85.62±0.47**	**80.08±0.46**	**79.56±0.25**	**74.08±0.47**	**87.96±0.02**	<u>91.47±0.44</u>	**84.38±0.25**	**81.88±0.46**
% Improvement (up to)	9.74%	10.11%	7.91%	15.16%	12.12%	4.37%	9.97%	6.74%

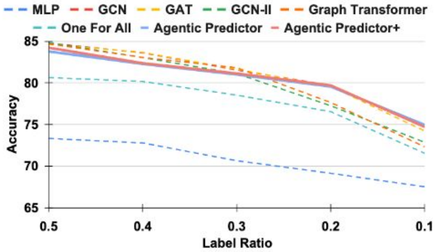
Experiments: ***Results*** *(Effects of Pretraining (Q2))*



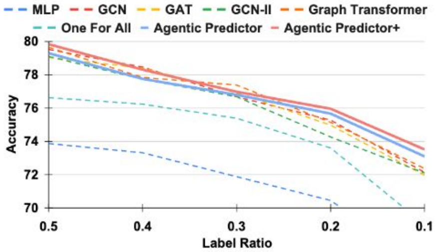
(a) Code Generation.



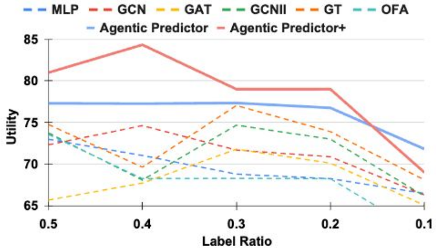
(b) Math Problem.



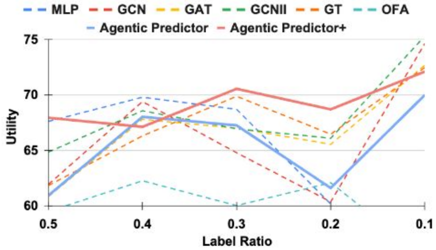
(c) Reasoning Tasks.



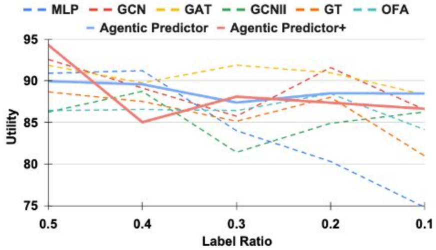
(d) Average.



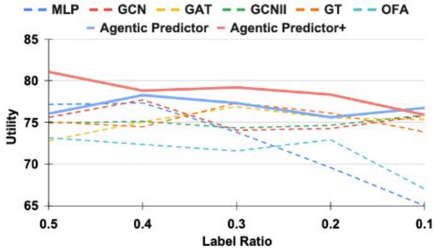
(e) Code Generation.



(f) Math Problem.



(g) Reasoning Tasks.



(h) Average.

Figure 4. Comparison of accuracy (upper) and utility (lower) between Agentic Predictor and the baselines across varying label ratios.

Experiments: ***Results*** *(Ablation Study (Q3))*

Table 4. Results of ablation study on different input view variations.

View Variations			Code Generation		Math Problem		Reasoning Task		Average	
Code	Graph	Text	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility
✓			82.04±0.51	75.66±0.66	75.70±0.14	68.52±0.91	83.19±0.56	**91.51±0.61**	80.31±0.40	78.56±0.73
	✓		84.44±0.31	77.22±3.46	79.14±0.28	67.99±3.36	87.00±0.21	91.03±1.23	83.53±0.27	78.75±2.68
		✓	79.87±0.28	70.34±0.43	76.60±0.65	68.45±1.80	68.06±0.00	71.04±0.00	74.84±0.31	69.94±0.74
✓	✓		83.72±0.83	73.97±0.81	75.86±0.85	70.18±1.64	86.88±0.14	86.14±4.62	82.15±0.61	76.76±2.36
✓		✓	82.27±0.63	77.28±1.12	76.03±0.14	66.66±4.18	54.17±0.00	53.21±0.00	70.82±0.26	65.72±1.77
	✓	✓	82.45±1.36	74.64±1.57	75.70±1.26	67.83±3.71	69.47±0.00	70.55±0.00	75.87±0.87	71.01±1.76
✓	✓	✓	**85.62±0.47**	**80.08±0.46**	**79.56±0.25**	**74.08±0.47**	**87.96±0.02**	91.47±0.44	**84.38±0.25**	**81.88±0.46**

Table 5. Results of ablation study on different input graph variations.

Graph Variations		Code Generation		Math Problem		Reasoning Task		Average	
Single Graph	Multi Graph	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility	Accuracy	Utility
✓		82.58±0.54	78.52±2.91	78.57±0.73	67.51±2.11	86.95±0.13	90.14±3.10	82.70±0.47	78.72±2.71
	✓	84.44±0.31	77.22±3.46	79.14±0.28	67.99±3.36	87.00±0.21	91.03±1.23	83.53±0.27	78.75±2.68

Experiments: *Downstream Performance*

Table 6. Workflow optimization performance based on the selected workflow across different methods.

Methods	Math Problems		Code Generation		Reasoning			Average	
	MATH	GSM8K	MBPP	HumanEval	MMLU	DROP	HotpotQA	Score	Search Cost ($)
Ground Truth (AFlow)	87.38	94.53	73.22	97.20	83.10	84.25	69.94	84.23	39.83
Random	78.40	75.23	67.84	76.34	42.87	80.42	16.86	62.56	0.00
GCN	79.22	86.16	68.23	97.46	46.43	82.33	19.14	68.42	0.00
GAT	80.11	86.22	**68.62**	97.71	57.00	85.83	**21.47**	71.00	0.00
Agentic Predictor	**81.89**	**92.65**	68.42	**98.73**	**79.70**	**86.25**	13.37	**74.43**	0.00

Conclusions

- We propose **Agentic Predictor**, a lightweight, predictive framework to estimate the success of agentic workflows using multi-view representation learning and unsupervised pretraining.
- We use the multi-view encoding to capture **workflow heterogeneity** from
    - Graph Structure (agent interaction),
    - Code Semantics (logic & tool use), and
    - Instruction Prompts (roles & behaviors).
- We introduce cross-domain unsupervised pretraining to lessen the **label scarcity** problem by training the encoder on unlabeled workflows from various domain.

Thank you!

GitHub: github.com/deepauto-ai/agentic-predictor
Email: patara@deepauto.ai