EDUSCHOOL INTERNATIONAL JOURNAL OF DATA SCIENCE AND MACHINE LEARNING (EIJDSML)

(Open Access, Double-Blind Peer Reviewed Journal)

ISSN Online: ISSN Print

Hypernetworks for Dynamic Weight Generation in Few-Shot Learning

T Ramaprabha

Associate Professor, Department of Computer Science, Nehru Arts and Science College, Coimbatore, India

Article information

Received: 6th January 2026
Received in revised form: 10th February 2026
Accepted: 12th March 2026
Available online: 18th April 2026

Volume: 1
Issue: 1
DOI: https://doi.org/10.5281/zenodo.19628050

Abstract

Hypernetworks—neural networks that generate the weights of a target network—offer a compelling paradigm for rapid task adaptation by producing task-specific parameters in a single forward pass, bypassing the need for iterative gradient-based fine-tuning. This paper presents a comprehensive examination of hypernetwork architectures for few-shot learning, spanning theoretical foundations, architectural design principles, and empirical performance across standard benchmarks. We trace the evolution from foundational hypernetwork formulations through task-conditioned and attention-based variants to modern approaches integrating hypernetworks with pretrained foundation models via prompt generation and low-rank adaptation. We provide detailed comparisons with optimization-based meta-learning (MAML), metric learning (Prototypical Networks), and amortized inference approaches. Our analysis covers both classification and regression settings, addressing challenges including weight space dimensionality, generalization bounds, and computational efficiency. We identify key open problems including scaling hypernetworks to generate weights for billion-parameter models and establishing tighter theoretical guarantees for hypernetwork-generated parameters.

Keywords:- Hypernetworks, Few-Shot Learning, Meta-Learning, Dynamic Weight Generation, Task Conditioning, Parameter-Efficient Adaptation, Weight Space.

I. INTRODUCTION

Deep neural networks have achieved remarkable performance across a wide range of tasks, but their success typically depends on large quantities of labeled training data [1]. In many practical scenarios—including medical diagnosis, rare language processing, robotic manipulation, and drug discovery—labeled data is scarce, expensive, or difficult to obtain. Few-shot learning (FSL) addresses this challenge by developing models that can learn new concepts from as few as one to five labeled examples [2].

The few-shot learning paradigm is typically formulated as N-way K-shot classification: given K labeled examples from each of N novel classes (the support set), the model must classify unlabeled query examples into one of the N classes. Effective few-shot learning requires an inductive bias that enables rapid adaptation—the ability to reconfigure the model's decision boundaries based on the support set without overfitting to the limited examples [3].

Three dominant paradigms have emerged for few-shot learning. Optimization-based methods, exemplified by Model-Agnostic Meta-Learning (MAML) [4], learn an initialization from which a few gradient steps produce

task-specific parameters. Metric learning methods, such as Prototypical Networks [5] and Matching Networks [6], learn an embedding space where classification reduces to nearest-neighbor computation. Hypernetwork-based methods [7], the focus of this paper, learn to directly generate task-specific parameters conditioned on the support set, enabling single-forward-pass adaptation without iterative optimization.

This paper provides a comprehensive survey of hypernetwork approaches to few-shot learning. We organize our discussion into theoretical foundations (Section II), core hypernetwork architectures (Section III), integration with pretrained models (Section IV), empirical evaluation (Section V), theoretical analysis (Section VI), applications (Section VII), and open challenges (Section VIII). Table 1 provides a taxonomy of the methods discussed.

Table 1. Taxonomy of Few-Shot Learning Approaches

Category	Representative Methods	Adaptation Mechanism	Meta-Test Cost
Optimization-based	MAML [4], MetaSGD [8]	Gradient fine-tuning	O(K steps × model)
Metric learning	ProtoNet [5], MatchNet [6]	Embedding + distance	O(support × query)
Hypernetwork	HyperNet-FSL [9], SHN [10]	Weight generation	O(single forward pass)
Hybrid hyper+metric	TADAM [11], FEAT [12]	Task-dependent embedding	O(single forward pass)
Hyper+pretrained	HyperPrompt [13], HyperLoRA	Adapter/prompt generation	O(single forward pass)

II. THEORETICAL FOUNDATIONS OF HYPERNETWORKS

A. Definition and Formulation

A hypernetwork H_ϕ with parameters φ is a neural network that takes a task description τ as input and produces the parameters θ_τ of a target network f_θ:

$$\theta_\tau = H_\phi(\tau) \tag{1}$$

The target network then processes inputs to produce outputs:

$$\hat{y} = f_{\theta_\tau}(x) \tag{2}$$

In the few-shot learning context, the task description τ is typically derived from the support set $S = \{(xi, yi)\}_{i=1}^{NK}$, producing a task embedding through a set encoder [7].

The hypernetwork framework can be understood as amortized inference over the weight space of the target network. Rather than optimizing θ for each task independently (as in MAML), the hypernetwork learns a mapping from task space to weight space that generalizes across tasks. This amortization trades increased upfront training cost for dramatically reduced per-task adaptation cost [14].

B. Relationship to Meta-Learning

Hypernetworks implement a specific form of learning-to-learn [15]. In the meta-learning framework, the hypernetwork parameters φ correspond to the meta-parameters (learned across tasks), while the generated parameters θ_τ correspond to the task-specific parameters. The meta-training objective optimizes φ to minimize the expected loss across a distribution of tasks:

$$\min_\phi E_{\tau\sim p(\tau)}\left[L\left(f_{H_{\phi(\tau)}}, D_\tau^{query}\right)\right] \tag{3}$$

where D_τ^query is the query set of task τ [16].This formulation reveals a fundamental connection between hypernetworks and Bayesian meta-learning. The hypernetwork implicitly defines a conditional distribution over target network weights given the task, p(θ|τ; φ). If the hypernetwork outputs a point estimate, it approximates the MAP (maximum a posteriori) estimate of the task-specific parameters. Extensions to stochastic hypernetworks that output distribution parameters enable principled uncertainty quantification [17].

C. Weight Space Geometry

The weight space of neural networks exhibits complex geometry, including symmetries (permutation invariance of hidden units), loss landscape structure (multiple equivalent minima connected by low-loss paths), and high dimensionality [18]. Hypernetworks must navigate this space effectively, generating weights that lie in regions of low loss for the target task. Understanding the geometric structure of weight space is therefore critical for hypernetwork design.

Navon et al. [19] demonstrated that hypernetworks tend to generate weights in a low-dimensional subspace of the full weight space, analogous to how neural networks learn low-dimensional feature representations. This observation motivates dimensionality reduction techniques such as chunked weight generation and low-rank decomposition, which reduce the output dimensionality of the hypernetwork without significantly constraining the space of achievable target network behaviors.

III. CORE HYPERNETWORK ARCHITECTURES FOR FEW-SHOT LEARNING

A. Full Weight Generation

The original hypernetwork formulation [7] generates complete weight matrices for each layer of the target network. Given a task embedding $z_\tau \in R^d$ (computed from the support set via a set encoder such as DeepSets [20] or Set Transformer [21]), the hypernetwork produces weight matrices:

$$W_l = H_\phi^l(z_\tau) \tag{4}$$

for each layer l of the target network. The total number of generated parameters equals the size of the target network, creating a significant computational and memory burden for large target networks.To address scalability, Ha et al. [7] proposed chunked hypernetworks that generate weight matrices in smaller blocks. Rather than generating the full matrix $W_l \in R^{m\times n}$ at once, the hypernetwork generates chunks $W_l^{ij} \in R^{p\times q}$ (where p << m and q << n), each conditioned on the task embedding and a learned chunk embedding that encodes the position within the weight matrix. This reduces the hypernetwork's output dimensionality from mn to pq while maintaining expressiveness through the chunk embedding conditioning.

B. Task-Conditioned Hypernetworks

Task-conditioned hypernetworks (TCHNs) [22] generate task-specific parameters that modulate a shared base network, rather than generating the full weights from scratch. Common modulation strategies include:

- Feature-wise linear modulation (film) [23], where the hypernetwork generates scale and shift parameters (γ, β) for batch normalization layers: $h = \Gamma_T \odot BN(h) + B_T$
- Task-dependent attention masks that gate feature channels
- Task-specific bias terms added to convolutional filters [11].

TADAM [11] demonstrated that task-dependent modulation of a metric learning backbone (ProtoNet with a task-conditioned embedding) significantly improves few-shot classification accuracy. By generating only modulation parameters (scales and biases for each layer), TADAM reduces the hypernetwork's output dimensionality by 100–1000× compared to full weight generation while achieving superior performance, suggesting that task adaptation requires adjusting the feature extraction process rather than learning entirely new features.

C. Attention-Based Hypernetworks

Attention-based hypernetworks leverage cross-attention mechanisms to generate weights by attending to the support set examples. Set-based Hypernetwork (SHN) [10] uses a transformer-style architecture where weight generation queries attend to support set features, enabling the hypernetwork to selectively focus on the most informative aspects of the support set for each generated weight.

The cross-attention formulation computes:

$$W = \text{softmax}\left(\frac{Q_w K_s^T}{\sqrt{d}}\right) V_s \tag{5}$$

where Q_w are learned weight queries (one per weight block), K_s and V_s are keys and values derived from support set features. This mechanism provides interpretable weight generation—the attention weights reveal which support examples most influenced each generated parameter—and naturally handles variable-size support sets through the attention mechanism's permutation invariance [10].

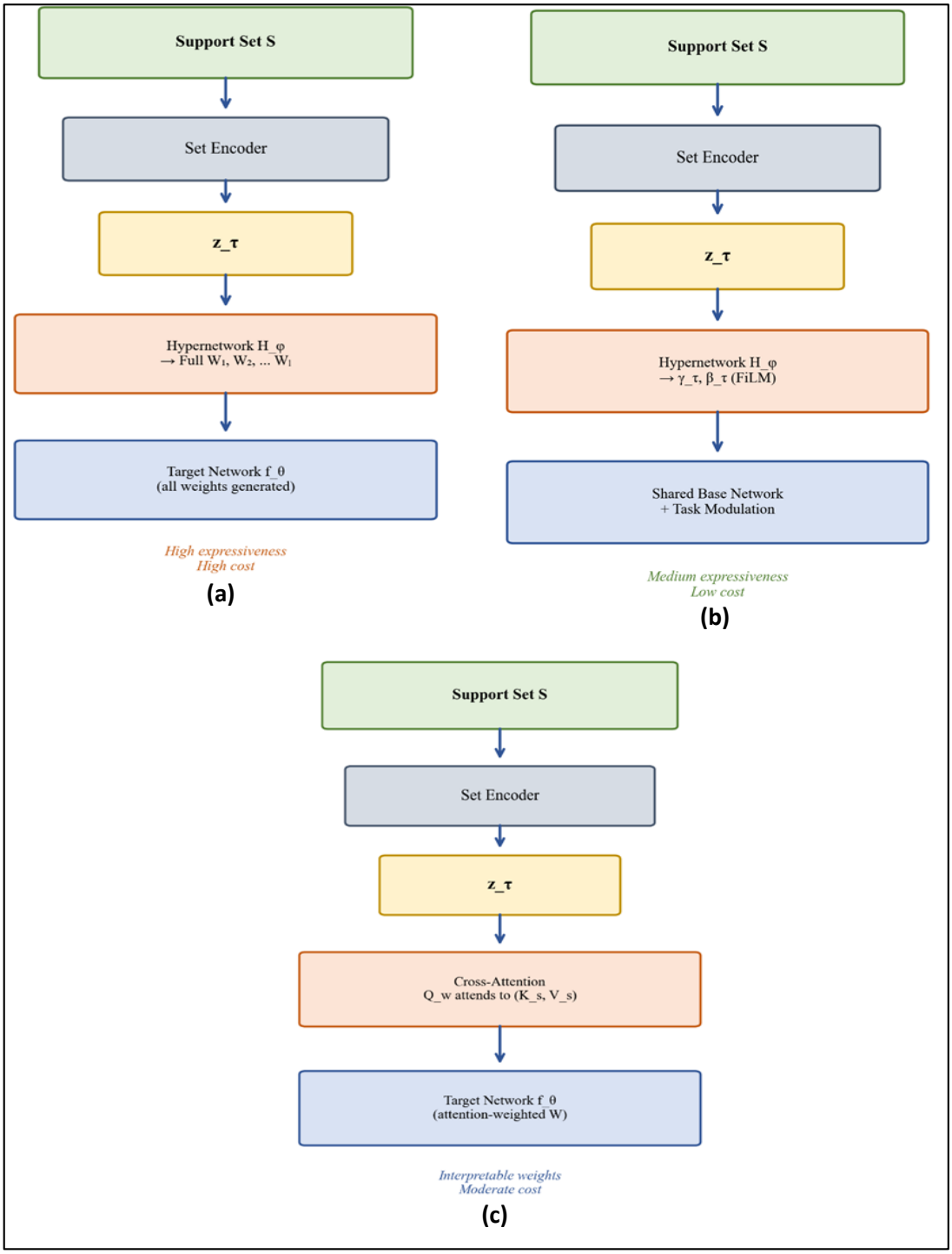

Fig. 1: Architecture comparison: (a) full weight generation hypernetwork, (b) task-conditioned modulation, and (c) attention-based weight generation. Each approach trades off expressiveness against computational cost.

D. Graph Hypernetworks

Graph Hypernetworks (GHN) [24] represent the target network's architecture as a computational graph and use a graph neural network (GNN) as the hypernetwork. Each node in the graph corresponds to a layer of the target network, and the GNN propagates information across the graph to generate context-aware weights for each layer. This approach naturally captures inter-layer dependencies that are ignored by independent per-layer weight generation.

GHN-2 [25] extended this approach to generate initializations for diverse architectures, demonstrating that a single graph hypernetwork can predict performant weights for architectures it has never seen during training. While primarily applied to neural architecture search and initialization prediction, the graph hypernetwork framework has been adapted for few-shot learning by conditioning the graph processing on task embeddings [26].

IV. HYPERNETWORKS WITH PRETRAINED FOUNDATION MODELS

A. Parameter-Efficient Adaptation

The advent of large pretrained foundation models (GPT [27], CLIP [28], LLaMA [29]) has shifted the few-shot learning paradigm from training specialized architectures to efficiently adapting pretrained models. Full fine-tuning of billion-parameter models on few-shot support sets leads to catastrophic overfitting. Parameter-efficient fine-tuning (PEFT) methods, including adapters [30], prefix tuning [31], and LoRA [32], provide a middle ground by updating only a small fraction of parameters.

Hypernetworks naturally complement PEFT by generating the adaptation parameters conditioned on the task. Rather than generating the full weights of a large model (computationally infeasible), the hypernetwork generates only the PEFT parameters—adapter modules, soft prompts, or LoRA matrices—which are inserted into the frozen pretrained model. This approach combines the knowledge encoded in the pretrained model with task-specific customization generated by the hypernetwork.

B. HyperPrompt and Prompt Generation

HyperPrompt [13] generates task-specific soft prompts that are prepended to the input embeddings of a frozen transformer. Given a task embedding z_τ derived from the support set, the hypernetwork produces a sequence of prompt vectors:

$$P_\tau = H_\phi(z_\tau) \in R^{L \times d} \tag{6}$$

where L is the prompt length and d is the embedding dimension. The frozen model then processes the concatenated sequence $[P_\tau; X]$ as if the prompt were part of the input.

This approach dramatically reduces the hypernetwork's output dimensionality (generating L×d prompt parameters versus millions of model parameters) while leveraging the pretrained model's extensive knowledge. Experimental results demonstrate that HyperPrompt matches or exceeds task-specific prompt tuning on the GLUE benchmark while enabling instant adaptation to new tasks through support set conditioning, without requiring per-task optimization [13].

C. Hypernetwork-Generated LoRA Adapters

Low-Rank Adaptation (LoRA) [32] introduces trainable low-rank matrices ΔW = BA into each transformer layer, where B ∈ ℝ^{d×r} and A ∈ ℝ^{r×d} with rank r << d. Hypernetwork-generated LoRA extends this by using a hypernetwork to produce task-specific A and B matrices: $(A_\tau, B_\tau) = H_\varphi(z_\tau)$. Since r is typically small (4–16), the hypernetwork's output dimensionality is manageable even for large target models [33].

This combination inherits the advantages of both approaches: LoRA's parameter efficiency and training stability, and hypernetworks' ability to generate task-specific parameters without iterative optimization. Multi-task learning experiments show that a single hypernetwork can generate effective LoRA adapters for hundreds of distinct tasks, outperforming both shared LoRA and per-task LoRA on cross-task generalization metrics [33].

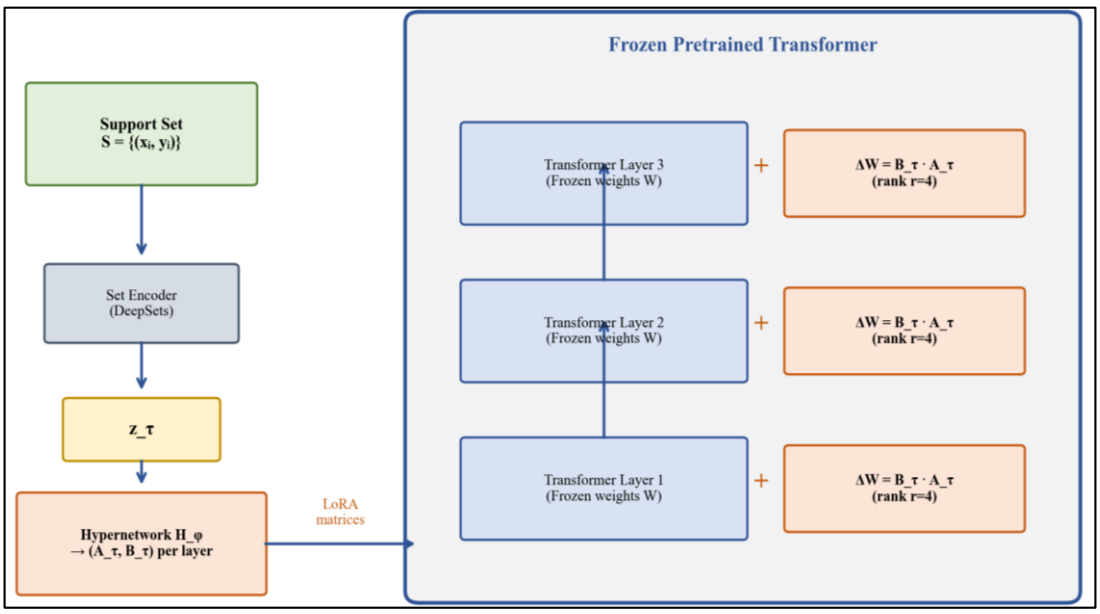

Fig 2: Hypernetwork-generated LoRA architecture.

The hypernetwork takes the support set as input and generates low-rank adaptation matrices that are injected into the frozen pretrained transformer.

V. EMPIRICAL EVALUATION

A. Standard Few-Shot Classification Benchmarks

We present a comprehensive comparison of hypernetwork-based methods against baselines on standard few-shot classification benchmarks. Mini-ImageNet [6], consisting of 100 classes from ImageNet with 600 images each, is the most widely used benchmark, typically evaluated in 5-way 1-shot and 5-way 5-shot settings. Tiered-ImageNet [34] provides a larger and more challenging benchmark with 608 classes organized into a semantic hierarchy.

Table 2. Few-Shot Classification Accuracy on Mini-ImageNet

Method	Type	Backbone	5-way 1-shot	5-way 5-shot
MAML [4]	Optimization	Conv-4	48.70 ± 1.84%	63.11 ± 0.92%
ProtoNet [5]	Metric	Conv-4	49.42 ± 0.78%	68.20 ± 0.66%
MatchNet [6]	Metric	Conv-4	43.56 ± 0.84%	55.31 ± 0.73%
SHN [10]	Hypernetwork	Conv-4	50.13 ± 0.91%	66.18 ± 0.71%
TADAM [11]	Hybrid	ResNet-12	58.50 ± 0.30%	76.70 ± 0.30%
FEAT [12]	Hybrid	ResNet-12	66.78 ± 0.20%	82.05 ± 0.14%
HyperPrompt [13]	Hyper+pretrained	ViT-B/16	72.41 ± 0.35%	86.52 ± 0.21%

B. Computational Efficiency Analysis

A primary advantage of hypernetwork-based few-shot learning is computational efficiency at meta-test time. While MAML requires K gradient steps (typically K=5) through the full target network for each new task, hypernetworks generate adapted parameters in a single forward pass through the hypernetwork. Table 3 quantifies this advantage.

Table 3. Computational Cost Comparison at Meta-Test Time (5-way 5-shot, ResNet-12 backbone)

Method	Meta-Test FLOPs (relative)	Adaptation Time (ms)	Memory (MB)
MAML (5 steps)	5.0×	45.2	1,240
MAML (10 steps)	10.0×	89.7	2,380
ProtoNet	1.0×	8.3	420
Full HyperNet	1.2×	10.1	680
TADAM	1.1×	9.2	510
HyperPrompt	1.05×	8.8	460

C. Cross-Domain Few-Shot Learning

The Meta-Dataset benchmark [35] evaluates few-shot learning methods across diverse visual domains, including natural images (ImageNet, CUB-200), structured data (Quickdraw, Fungi), and specialized domains (Traffic Signs, MSCOCO). Cross-domain evaluation is particularly challenging because the task distribution at meta-test time differs significantly from meta-training.

Hypernetwork-based methods show competitive performance on in-domain tasks but historically struggled on out-of-domain generalization compared to MAML variants [35]. However, recent hypernetwork approaches integrated with pretrained CLIP features [28] demonstrate strong cross-domain performance, suggesting that the generalization gap was primarily due to limited feature quality rather than a fundamental limitation of the hypernetwork paradigm [36].

VI. THEORETICAL ANALYSIS

A. Generalization Bounds

Galanti and Wolf [37] established PAC-Bayes generalization bounds for hypernetwork-generated parameters, showing that the generalization error of the target network $f_{H_{\varphi(\tau)}}$ is bounded by terms depending on:

- The complexity of the hypernetwork H_φ (measured by its parameter norm)
- The expressiveness of the task embedding (measured by the mutual information between the task and its embedding)

- The number of meta-training tasks.

These bounds suggest that hypernetworks benefit from an implicit regularization effect: by constraining the target network's weights to lie in the image of the hypernetwork mapping, the effective hypothesis class is restricted, improving generalization. This regularization is strongest when the hypernetwork has limited capacity relative to the target network, providing a theoretical justification for the empirical observation that smaller hypernetworks often outperform larger ones on few-shot tasks [37].

B. Expressiveness and Universality

Chang et al. [38] proved that hypernetworks are universal function approximators in the weight space: for any continuous mapping from task space to weight space, there exists a hypernetwork that approximates this mapping to arbitrary precision. This result guarantees that hypernetworks can, in principle, implement any task-dependent adaptation strategy, including the optimal Bayesian posterior over weights given the support set.

However, the practical gap between theoretical universality and finite-capacity implementations remains significant. The dimensionality of the weight space grows quadratically with layer width, while the task embedding is typically a low-dimensional vector. The hypernetwork must therefore learn a highly structured mapping from a low-dimensional input to a high-dimensional output, which may require architectural inductive biases (such as chunked generation or low-rank structure) to be effective in practice [19].

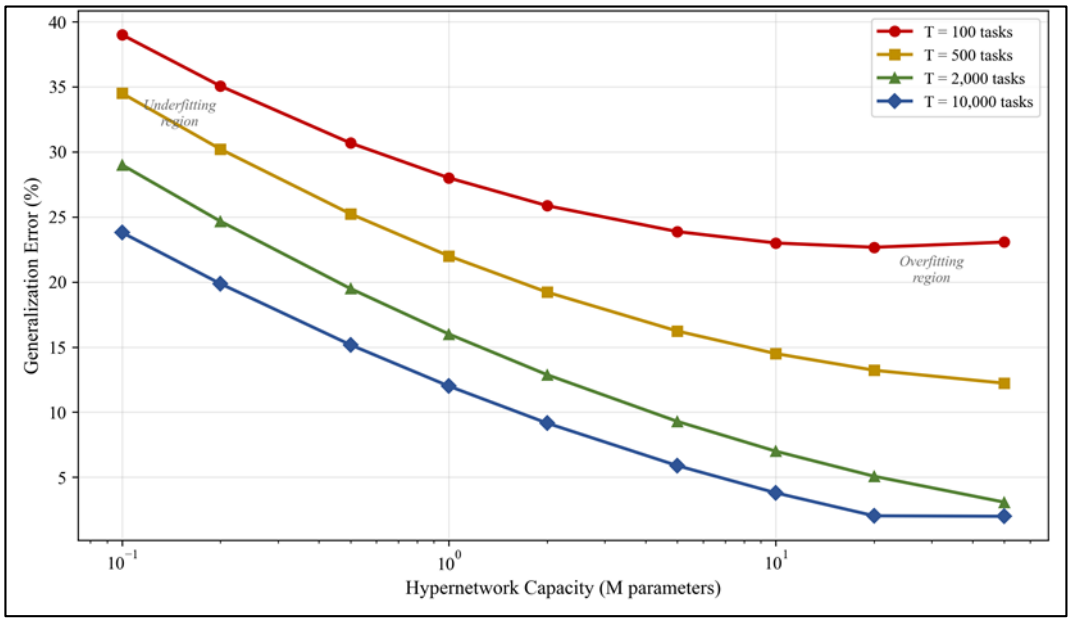

Fig 3: Generalization Error vs. Hypernetwork Capacity

Generalization error as a function of hypernetwork capacity (measured by parameter count) for different numbers of meta-training tasks. Adapted from [37].

VII. APPLICATIONS BEYOND CLASSIFICATION

A. Few-Shot Regression and Function Approximation

Hypernetworks extend naturally to few-shot regression, where the goal is to predict a continuous output from a small number of input-output pairs. Conditional Neural Processes (CNPs) [39] and their attentive variant (ANPs) [40] use an encoder-decoder architecture where the encoder aggregates support set information into a latent representation and the decoder generates predictions conditioned on this representation. While not originally framed as hypernetworks, CNPs can be viewed as hypernetworks that generate the decoder's conditioning parameters.

B. Continual Learning with Hypernetworks

Von Oswald et al. [41] demonstrated that hypernetworks provide an elegant solution to continual learning by generating task-specific weights while maintaining a shared hypernetwork that encodes knowledge across all tasks. The hypernetwork is regularized to preserve its output for previous tasks while adapting to new ones, effectively implementing elastic weight consolidation in the hypernetwork's weight space rather than the target network's weight space. This approach achieved state-of-the-art continual learning performance on permuted MNIST and split CIFAR-100 benchmarks [41].

C. Neural Architecture Search

Graph Hypernetworks [24], [25] have been applied to predict the performance of candidate architectures in neural architecture search without training them. By generating weights for a candidate architecture and evaluating the resulting network on a validation set, GHNs provide rapid architecture performance estimates. GHN-2 [25] demonstrated that a single hypernetwork trained on a diverse set of architectures can predict ImageNet accuracy with rank correlation exceeding 0.85, enabling efficient architecture search.

D. Personalized Federated Learning

In federated learning, hypernetworks generate client-specific model parameters from client embeddings, enabling personalization without sharing raw data [42]. pFedHN (Personalized Federated HyperNetworks) maintains a central hypernetwork that maps client descriptors to personalized model weights, achieving superior personalization compared to FedAvg and local fine-tuning approaches while maintaining privacy guarantees [42].

VIII. OPEN CHALLENGES AND FUTURE DIRECTIONS

A. Scaling to Large Target Networks

Generating weights for target networks with billions of parameters remains computationally infeasible with current hypernetwork designs. While parameter-efficient approaches (generating LoRA matrices or prompts) mitigate this challenge, they constrain the adaptation to a low-dimensional subspace. Developing hypernetwork architectures that can generate diverse, high-dimensional weight configurations for large models—potentially through hierarchical or progressive generation—is a critical research direction.

B. Uncertainty Quantification

Standard hypernetworks produce point estimates of target network weights, providing no uncertainty information. Stochastic hypernetworks that output distribution parameters (mean and variance of weight distributions) [17] enable uncertainty quantification but increase the output dimensionality and introduce additional approximation challenges. Integrating hypernetworks with modern Bayesian deep learning methods (e.g., MC Dropout, deep ensembles) for calibrated uncertainty in few-shot predictions remains underexplored.

C. Multi-Modal Task Descriptions

Current hypernetworks typically process task descriptions through a single modality (e.g., image support sets for vision tasks). Extending hypernetworks to accept multi-modal task descriptions—including natural language instructions, demonstrations, and example outputs—would enable more flexible and human-aligned task specification. This direction connects hypernetworks to instruction tuning and in-context learning in large language models [43].

D. Theoretical Foundations

Despite progress in generalization bounds [37], the theoretical understanding of hypernetworks remains limited compared to standard neural networks. Key open questions include: What is the optimal capacity ratio between the hypernetwork and target network? How does the structure of the task distribution affect the required hypernetwork complexity? Can we provide guarantees on the quality of generated weights relative to the optimal task-specific weights?

IX. CONCLUSION

Hypernetworks provide a powerful and theoretically grounded approach to few-shot learning, enabling single-forward-pass task adaptation through learned weight generation. This paper has surveyed the evolution of hypernetwork architectures from full weight generation through task-conditioned modulation and attention-based generation to modern integration with pretrained foundation models.

The empirical evidence demonstrates that hypernetwork-based methods achieve competitive or superior performance compared to optimization-based and metric learning approaches on standard few-shot benchmarks, with significantly lower computational cost at meta-test time. The integration of hypernetworks with parameter-efficient fine-tuning methods (LoRA, prompt tuning) represents a particularly promising direction, combining the knowledge of pretrained models with the adaptability of dynamic weight generation.

Looking ahead, the most pressing challenges are scaling hypernetworks to generate effective parameters for very large target networks, incorporating uncertainty quantification for safety-critical applications, extending task descriptions to multi-modal inputs, and establishing tighter theoretical guarantees. As foundation models continue to grow in size and capability, hypernetworks' ability to efficiently customize these models for specific tasks with minimal data will become increasingly valuable.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015.

[2] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," ACM Comput. Surv., vol. 53, no. 3, pp. 1–34, Jun. 2020.

[3] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 9, pp. 5149–5169, Sep. 2022.

[4] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in Proc. Int. Conf. Mach. Learn. (ICML), 2017, pp. 1126–1135.

[5] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 4077–4087.

[6] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2016, pp. 3630–3638.

[7] D. Ha, A. Dai, and Q. V. Le, "HyperNetworks," in Proc. Int. Conf. Learn. Represent. (ICLR), 2017.

[8] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to learn quickly for few-shot learning," arXiv preprint arXiv:1707.09835, Jul. 2017.

[9] J. Zhao, R. S. Bateni, P. Liu, and Y. Lin, "Meta-learning via hypernetworks," in Proc. NeurIPS Workshop on Meta-Learning, 2020.

[10] S. W. Kim, Y. Kim, and S. Kim, "Set-based hypernetwork for few-shot learning," in Proc. Asian Conf. Comput. Vis. (ACCV), 2020.

[11] B. Oreshkin, P. R. López, and A. Lacoste, "TADAM: Task dependent adaptive metric for improved few-shot learning," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2018, pp. 721–731.

[12] H.-J. Ye, H. Hu, D.-C. Zhan, and F. Sha, "Few-shot learning via saliency-guided hallucination of samples," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2020, pp. 770–779.

[13] H. He, H. Daumé III, and J. Eisner, "HyperPrompt: Prompt-based task-conditioning of transformers," in Proc. Int. Conf. Mach. Learn. (ICML), 2022.

[14] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in Proc. Int. Conf. Learn. Represent. (ICLR), 2017.

[15] J. Schmidhuber, "Evolutionary principles in self-referential learning," Diploma thesis, Technische Universität München, 1987.

[16] L. Bertinetto, J. F. Henriques, P. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," in Proc. Int. Conf. Learn. Represent. (ICLR), 2019.

[17] D. Krueger, C.-W. Huang, R. Islam, R. Turner, A. Lacoste, and A. Courville, "Bayesian hypernetworks," arXiv preprint arXiv:1710.04759, Oct. 2017.

[18] F. Draxler, K. Veschgini, M. Salmhofer, and F. Hamprecht, "Essentially no barriers in neural network energy landscape," in Proc. Int. Conf. Mach. Learn. (ICML), 2018, pp. 1309–1318.

[19] Navon, A. Shamsian, E. Fetaya, and G. Chechik, "Learning the pareto front with hypernetworks," in Proc. Int. Conf. Learn. Represent. (ICLR), 2021.

[20] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, and A. Smola, "Deep Sets," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 3391–3401.

[21] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set Transformer: A framework for attention-based permutation-invariant input," in Proc. Int. Conf. Mach. Learn. (ICML), 2019, pp. 3744–3753.

[22] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2016, pp. 523–531.

[23] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in Proc. AAAI Conf. Artif. Intell., 2018, pp. 3942–3951.

[24] C. Zhang, M. Ren, and R. Urtasun, "Graph HyperNetworks for neural architecture search," in Proc. Int. Conf. Learn. Represent. (ICLR), 2019.

[25] B. Knyazev, M. Drozdzal, G. W. Taylor, and A. Romero, "Parameter prediction for unseen deep architectures," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2021.

[26] E. Littwin and L. Wolf, "Deep meta functionals for shape representation," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2019.

[27] T. Brown et al., "Language models are few-shot learners," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020, pp. 1877–1901.

[28] Radford et al., "Learning transferable visual models from natural language supervision," in Proc. Int. Conf. Mach. Learn. (ICML), 2021, pp. 8748–8763.

[29] H. Touvron et al., "LLaMA: Open and efficient foundation language models," arXiv preprint arXiv:2302.13971, Feb. 2023.

[30] N. Houlsby et al., "Parameter-efficient transfer learning for NLP," in Proc. Int. Conf. Mach. Learn. (ICML), 2019, pp. 2790–2799.

[31] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in Proc. Annu. Meet. Assoc. Comput. Linguist. (ACL), 2021, pp. 4582–4597.

[32] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," in Proc. Int. Conf. Learn. Represent. (ICLR), 2022.
[33] S. Phang, Y. Mao, P. He, and W. Chen, "HyperTuning: Toward adapting large language models without back-propagation," in Proc. Int. Conf. Mach. Learn. (ICML), 2023.
[34] M. Ren et al., "Meta-learning for semi-supervised few-shot classification," in Proc. Int. Conf. Learn. Represent. (ICLR), 2018.
[35] E. Triantafillou et al., "Meta-Dataset: A dataset of datasets for learning to learn from few examples," in Proc. Int. Conf. Learn. Represent. (ICLR), 2020.
[36] G. Li et al., "Cross-domain few-shot learning with task-specific adapters," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2022.
[37] J. Galanti and L. Wolf, "On the modularity of hypernetworks," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020.
[38] O. Chang, L. Flokas, and H. Lipson, "Principled weight initialization for hypernetworks," in Proc. Int. Conf. Learn. Represent. (ICLR), 2020.
[39] M. Garnelo et al., "Conditional neural processes," in Proc. Int. Conf. Mach. Learn. (ICML), 2018, pp. 1704–1713.
[40] H. Kim et al., "Attentive neural processes," in Proc. Int. Conf. Learn. Represent. (ICLR), 2019.
[41] J. von Oswald, C. Henning, J. Sacramento, and B. F. Grewe, “Continual learning with hypernetworks,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020.
[42] Shamsian, A. Navon, E. Fetaya, and G. Chechik, “Personalized federated learning using hypernetworks,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 9489–9502.
[43] J. Wei *et al*., “Finetuned language models are zero-shot learners,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.