HyperTransformer

Timofey Chernikov

MIPT, 2023

November 6, 2023

Motivation

Problem statement

Given a broad set of tasks, develop a single model that will be able to quickly specialize for a particular task with little data.

Existing approaches

Metric-Based Learning

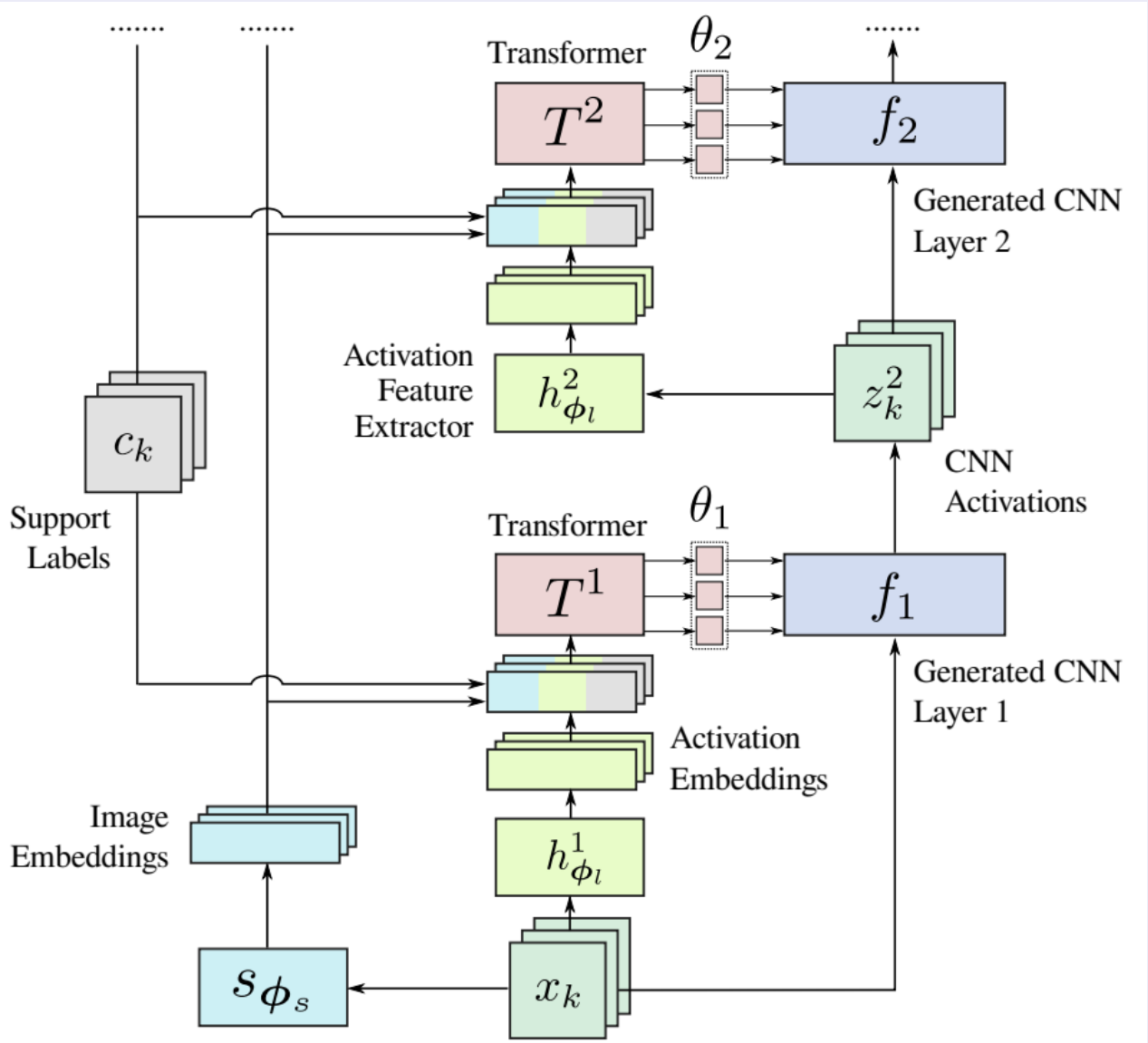
One family of approaches involves mapping input samples into an embedding space and then using some nearest neighbor algorithm to label query samples based on the distances from their embeddings to embeddings of labeled support samples. The metric used to compute the distances can either be the same for all tasks, or can be task-dependent

Formal definition

A set of tasks $\{t| \in \mathcal{T}\}$
Loss $\mathcal{L}(f, t)$ that quantifies the correctness of any model f attempting to solve t.
A task description $\tau(t)$ that is sufficient for communicating this task and finding the optimal model that solves it, includes any available information like labeled and unlabeled samples, image metadata, textual descriptions, etc.
The weight generation algorithm can then be viewed as a method of using a set of training tasks $\mathcal{T}_{train}$ for discovering a particular solver a that given $\tau(t)$ for a task t similar to those present in the training set, produces an optimal model $f* = a_\phi(\tau) \in \mathcal{F}$ minimizing $\mathcal{L}(f*, t)$

$$\underset{\phi in \Phi}{\arg\min}\, \mathbb{E}_{t\sim p(t)} \mathcal{L}(a_\phi(\tau(t)), t)$$

Model scheme



Model description

A solver a_ϕ is the core of a few-shot learning algorithm - it encodes the knowledge of the training task distribution within its weights ϕ. It's a Transformer-based model that takes a task description τ and produces weights for some or all layers $\{\theta_l | l \in [1, L]\}$ of the generated CNN model. Layer weights that are not generated are instead learned end-to-end together with HT weights as ordinary task-agnostic variables. In other words, these learned layers are modified during the training phase and remain static during the evaluation phase. In our experiments generated CNN models contain a set of convolutional layers and a final fully-connected logits layer. The weights are generated layer-by-layer starting from the first layer: $\theta_1(\tau) \to \theta_2(\theta_1; \tau) \to ... \to \theta_L(\theta_1, ..., \theta_{L-1}; \tau)$.

Results

MINIIMAGENET						TIEREDIMAGENET		
Method	**1-S**	**5-S**	**Method**	**1-S**	**5-S**	**Method**	**1-S**	**5-S**
HT	54.1	68.5	HT-48	**55.1**	68.1	HT-32	**54.0**	**70.2**
MN	43.6	55.3	SAML	52.2	66.5	MAML-32	51.7	**70.3**
IMP	49.2	64.7	GCR	53.2	**72.3**	HT	**56.3**	**73.9**
PN	49.4	68.2	KTN	54.6	71.2	PN	53.3	72.7
MELR	**55.4**	**72.3**	PARN	**55.2**	71.6	MELR	**56.4**	**73.2**
TAML	51.8	66.1	PPA	54.5	67.9	RN	54.5	71.3

Table 2. Comparison of MINIIMAGENET and TIEREDIMAGENET 1-shot (1-S) and 5-shot (5-S) 5-way results for HT (underlined) and other widely known methods with a 64-64-64-64 model including (Tian et al., 2020): Matching Networks (Vinyals et al., 2016), IMP (Allen et al., 2019), Prototypical Networks (Snell et al., 2017), TAML (Jamal & Qi, 2019), SAML (Hao et al., 2019), GCR (Li et al., 2019a), KTN (Peng et al., 2019), PARN (Wu et al., 2019), Predicting Parameters from Activations (Qiao et al., 2018), Relation Net (Sung et al., 2018), MELR (Fei et al., 2021). We also include results for CNNs with fewer channels ("-32" for 32-channel models, etc.).

Literature

1. **Main article** HyperTransformer: Model Generation for Supervised and Semi-Supervised Few-Shot Learning.