A. Additional Implementation Details

The hyperparameters for our models are listed in Tables 4 and 5. Key size d was 64 throughout all experiments. A dropout rate of 0.2 was applied to each layer of adaFFN. For the other adaptive models, the input dropout rate was set to 0.2 or 0.0. The dropout for the last two layers were varied as shown in Table 4 and 5. Due to memory constraints, we used adaLSTM with a smaller number of hidden units (i.e., 200 vs 300) for deep models when applying to 3-shot tasks.

The neural network weights were initialized using He et al. (2015)'s method. We set the hard gradient clipping threshold for adaCNN model to 10. No gradient clipping was performed for the other models. We listed the setup for optimizers in Table 4 and 5. For Adam optimizer, the rest of the hyperparameters were set to their default values (i.e., $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$).

Although different parameterizations for the meta learner function g may improve the performance, for simplicity we used a 3-layer MLP with ReLU activation with 20 or 40 units per layer. This MLP acts coordinate-wise and processes conditioning information for each neuron independently.

Empirically, we found that selecting the vector from V_t corresponding to the key k'_i with maximum cosine similarity to the query k_j (hard attention) gave similar performance to soft attention.

We occasionally observed difficulty in optimizing the LSTM+adaFFN models, often seeing no improvement in the training loss from certain initializations. Decreasing the learning rate and in case of DF information applying dropouts to adaFFN layers helped training this model.

Models were implemented using the Chainer (Tokui et al., 2015) framework[4].

Table 4. Hyperparameters for few-shot image classification tasks

Model	Layers	Filters	Dropout rate	Optimizer
adaCNN (∇)	5	32/64	0.0, 0.3, 0.3	Adam (α=0.001)
adaCNN (DF)	5	32/64	0.2, 0.3, 0.3	Adam (α=0.001)
adaResNet (∇)	4	64, 96, 128, 256	0.2, 0.5, 0.5	SGD with momentum (lr=0.01, m=0.9)
adaResNet (DF)	4	64, 96, 128, 256	0.2, 0.5, 0.5	SGD with momentum (lr=0.01, m=0.9)

Table 5. Hyperparameters for few-shot language modelling tasks

Model	Hidden unit size	Dropout rate	Optimizer
2-layer LSTM + adaFFN (∇)	300	-	Adam (α=0.0003)
2-layer LSTM + adaFFN (DF)	300	0.2, 0.2, 0.2	Adam (α=0.0003)
1-layer adaLSTM (∇)	300	-	Adam (α=0.001)
1-layer adaLSTM (DF)	300	-	Adam (α=0.001)
2-layer adaLSTM (∇)	300, 200	-	Adam (α=0.001)
2-layer adaLSTM (DF)	300, 200	-	Adam (α=0.001)

[4]https://chainer.org/

B. Running Time Comparison with MetaNet

We compared the speed of our adaCNN model variants with MetaNet model on Mini-ImageNet task. We implemented all models in the Chainer framework (Tokui et al., 2015) and tested on an Nvidia Titan X GPU. In Figure 4 we see that adaCNN variants are significantly faster than MetaNet while being conceptually simpler and easier to implement.

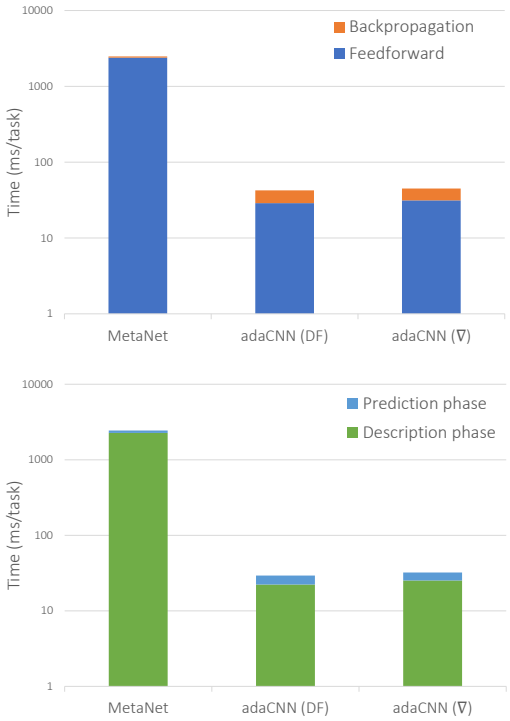

Figure 4. Training (top) and inference (bottom) speeds of MetaNet, adaCNN variants are compared on the 1-shot, 5-way Mini-ImageNet task. The y-axis shows wall-clock time (ms/task) in log scale. Training time includes the feedforward computations and the parameter updates. Inference time includes computations for the description and prediction phases.