Cross-Modulation Networks For Few-Shot Learning

Hugo Prol[1]**, Vincent Dumoulin**[2]**, and Luis Herranz**[1]
[1]Computer Vision Center, Universitat Autònoma de Barcelona
[2]Google Brain
hugo.prol.pereira@gmail.com, vdumoulin@google.com, lherranz@cvc.uab.es

Abstract

A family of recent successful approaches to few-shot learning relies on learning an embedding space in which predictions are made by computing similarities between examples. This corresponds to combining information between support and query examples at a very late stage of the prediction pipeline. Inspired by this observation, we hypothesize that there may be benefits to combining the information at various levels of abstraction along the pipeline. We present an architecture called *Cross-Modulation Networks* which allows support and query examples to interact throughout the feature extraction process via a feature-wise modulation mechanism. We adapt the Matching Networks architecture to take advantage of these interactions and show encouraging initial results on miniImageNet in the 5-way, 1-shot setting, where we close the gap with state-of-the-art.

1 Introduction

Recent deep learning successes in areas such as image recognition [1, 2, 3], machine translation [4, 5], and speech synthesis [6] rely on large amounts of data and extensive training. In contrast, humans excel in learning new concepts with very little supervision. Few-shot learning aims to close this gap by training models that can generalize well from few labeled examples.

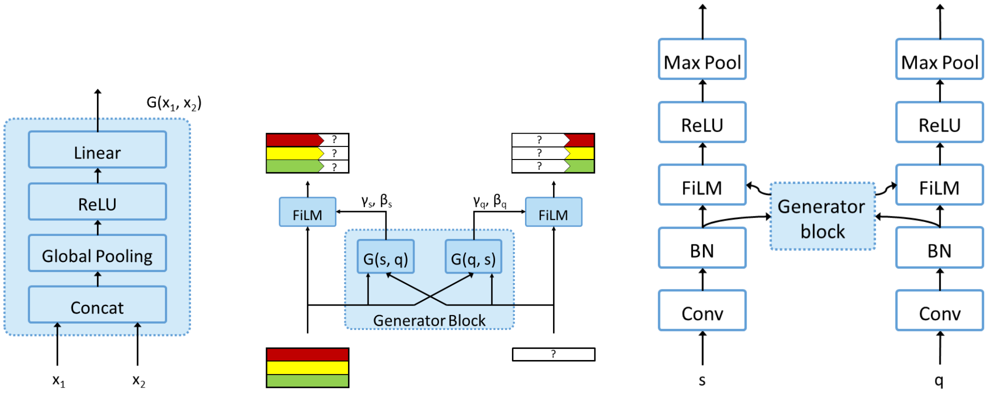
Several approaches have been proposed to tackle the few-shot learning problem, such as learning an initialization suitable to a small number of parameter updates when learning on a new problem with a small amount of data [7, 8, 9, 10, 11, 12], learning an embedding space in which examples are compared to make predictions [13, 14, 15, 16], learning the optimization algorithm which produces the final model [17], or learning a model which adapts to new problems through external memories or attention mechanisms [18, 19, 20, 21]. In particular, the approach which consists in learning an embedding space represents a simple yet effective solution to few-shot learning. This approach incorporates the inductive bias that examples should be compared at an abstract level of representation, which may be overly restrictive in cases where intermediate representations encode information useful to classification.

In this work we explore an extension applicable to metric learning architectures which allows the interaction of support and query examples at each level of abstraction. These interactions are implemented through a *cross-modulation mechanism* and enable the network to modify the intermediate features of the compared examples to produce better final representations and consequently a more robust metric space. Preliminary results on miniImageNet extending Matching Networks with this approach yield a performance comparable with state-of-the-art architectures of similar sizes.

2 Method

Few-shot learning and the episodic framework We adopt the *episodic approach* proposed by [13] and the nomenclature introduced in [22]. We partition classes into sets of C_{train} and C_{test}

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.



(a) FiLM generator (b) Cross-modulation mechanism (c) Final convolutional block

Figure 1: Architecture of the cross-modulation mechanism.

classes and train the model on K-shot, N-way episodes where the model learns from a *support set* $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{NK}$ containing K examples for each of the N classes (which are sampled from $\mathcal{C}_{train}$) and is required to generalize to a held-out *query set* $\mathcal{Q} = \{(\mathbf{x}_j^*, y_j^*)\}_{j=1}^{T}$. The model is evaluated on episodes constructed from $\mathcal{C}_{test}$.

Matching Networks In their simplest formulation, Matching Networks express $p(y^* \mid \mathbf{x}^*, \mathcal{S})$ by having each example in $\mathcal{S}$ cast a weighted vote, i.e.

$$p(y^* = c \mid \mathbf{x}^*, \mathcal{S}) = \sum_{i=1}^{NK} h(\mathbf{x}^*, \mathbf{x}_i) 1_{y_i = c}, \quad h(\mathbf{x}^*, \mathbf{x}_j) = \frac{\exp(\text{cosine}(f(\mathbf{x}^*), f(\mathbf{x}_i)))}{\sum_{j=1}^{NK} \exp(\text{cosine}(f(\mathbf{x}^*), f(\mathbf{x}_j)))}, \quad (1)$$

where $f(\cdot)$ is a parametrized feature extractor, and $h(\cdot, \cdot)$ computes the softmax over cosine similarities to the query example. Our implementation of $f(\cdot)$ follows the standard architecture used in the literature: a convolutional neural network with four blocks, each formed of a 3×3 convolution with 64 filters followed by batch normalization [23], a ReLU activation function, and a 2×2 max pooling operation.

Cross-modulation through FiLM layers Feature-wise Linear Modulation (FiLM) [24] allows to *modulate* the inner representation of a network by applying a feature-wise affine transformation. Let $x \in \mathbb{R}^{H \times W \times C}$ be the output of a convolutional layer for a given example, then a FiLM layer has the form $\text{FiLM}(x) = \gamma_z \odot x + \beta_z$, where $\odot$ denotes the Hadamard product and $\gamma_z, \beta_z \in \mathbb{R}^C$ are the *FiLM parameters*, computed by the *FiLM generator* $G(z)$ which takes the conditioning input z.

We introduce interactions between query and support examples using FiLM layers (see Figure 1b). The FiLM generator implements $G(x_1, x_2) = \varphi\left([x_1, x_2]\right) W + b$, where the weight matrix $W \in \mathbb{R}^{2C \times 2C}$ and bias $b \in \mathbb{R}^{2C}$ are learnable parameters, $[\cdot, \cdot]$ denotes channel-wise concatenation, and $\varphi : \mathbb{R}^{H \times W \times 2C} \to \mathbb{R}^{2C}$ represents global average pooling followed by ReLU (Figure 1a).

Our specific FiLM layer implementation is formulated as $\text{FiLM}(x) = (1 + \gamma_0 \gamma_z) \odot x + \beta_0 \beta_z$, as suggested by Oreshkin et al. [16]. We apply an L_1 regularization penalty to γ_0 and β_0 to enforce sparsity in the modulation, under the assumption that some features are more relevant than others in the modulation process. The input of the FiLM generator G is a batch of support-query example pairs resulting from the Cartesian product of the support and query batches. Note that in general $G(s_i, q_i) \neq G(q_i, s_i)$.

Previous work such as [16, 19] also makes use of feature-wise modulation in a few-shot learning context; our approach differentiates itself by considering pairwise interactions where both the support and the query example are used to predict the modulating parameters, and prediction happens locally at multiple levels of abstraction rather than via high-level features extracted from the support set.

Cross-Modulation Networks We extend Matching Networks by augmenting the feature extractor with our proposed cross-modulation mechanism. We use two embedding functions with shared

Table 1: Test accuracy on miniImageNet (%).

Model	**5-way 1-shot**	**5-way 5-shot**
Meta-Learner LSTM [17]	43.44 ± 0.77	60.60 ± 0.71
MAML [7]	48.70 ± 1.84	63.11 ± 0.92
Matching Networks[1] [13]	49.39 ± 0.62	66.16 ± 0.68
Prototypical Networks [14]	49.42 ± 0.78	**68.20 ± 0.66**
REPTILE [8]	49.97 ± 0.32	65.99 ± 0.58
PLATIPUS [10]	50.13 ± 1.86	-
Relation Nets [15]	**50.44 ± 0.82**	65.32 ± 0.70
Meta-SGD [26]	**50.47 ± 1.87**	64.03 ± 0.94
Cross-Modulation Nets	**50.94 ± 0.61**	66.65 ± 0.67

Table 2: Adding per-block noise in cross-modulation (5-way 1-shot on miniImageNet, ticks denoting noise introduction).

2	**3**	**4**	**Accuracy (%)**
			50.94 ± 0.61
✓			47.68 ± 0.58
	✓		49.28 ± 0.61
		✓	46.50 ± 0.58
✓	✓	✓	42.97 ± 0.56

parameters to encode separate batches of support and query examples. Their first convolutional block is identical to that of the baseline, while the other three blocks are cross-modulated, as illustrated in Figure 1c. We observed empirically that cross-modulation of the first block did not improve the results, and we decided not to cross-modulate it to save on computational overhead.

3 Experiments

Setup We experiment on the miniImageNet dataset [13] using the splits proposed by Ravi and Larochelle [17]. We train using the Adam optimizer [25]; the initial learning rate is set to 0.001 and halved every 10^5 episodes. For the *5-way, 1-shot* setup we use 15 query examples per episode when training Matching Networks but use 5 query examples per episode when training Cross-Modulation Networks, due to computational constraints. We set the L1 factor for the γ_0 and β_0 post-multipliers to 0.001 following cross-validation. Test accuracies are averaged over 1000 episodes using 15 query examples per episode, and we report 95% confidence intervals.

Cross-modulation helps increase test accuracy Table 1 compares our proposed approach with results published for comparable network architectures. We first note that our Matching Networks implementation exhibits a stronger accuracy ($49.39 \pm 0.62\%$) than what is usually reported, which is due to our use of the "unnormalized" cosine similarity used in the original implementation,[2] i.e.

$$\mathrm{cosine}_{\mathrm{u}}(\mathbf{x}^*, \mathbf{x}_i) = \mathbf{x}^* \cdot \mathbf{x}_i \|\mathbf{x}^*\|^{-1}. \quad (2)$$

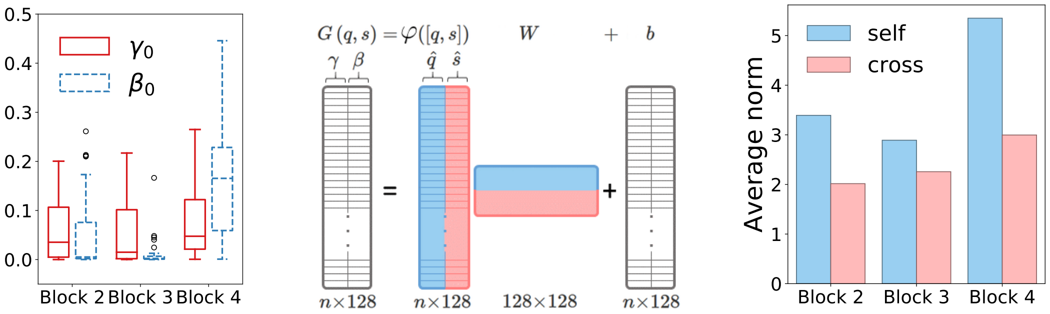
We hypothesize this is directly linked to the metric scaling properties discussed in [16].

Applying our proposed cross-modulation architecture to the baseline Matching Networks model trained in the *5-way, 1-shot* setting increases its accuracy to $50.94 \pm 0.61\%$ — a 1.55% increase over the corresponding baseline — and places it on-par with other SOTA approaches. In the *5-way, 5-shot* setting, our implementation of Matching Networks is also very competitive, and augmenting it with our proposed cross-modulation mechanism appears to improve its accuracy, albeit more modestly (0.59%). It could be that cross-modulation is best suited to the very low-data regime and offers diminishing returns as the number of examples per class increases, but a more thorough investigation is required to draw definitive conclusions. Future work can also address the design of more efficient and effective ways to exploit interactions in the few-shot case.

The model takes advantage of cross-modulation Table 1 shows an improvement over the Matching Networks baseline, which we attribute to the interaction between support and query examples via our proposed cross-modulation mechanism. To verify that this interaction is necessary for the performance of the trained network, we compare the test accuracy of the model in normal conditions to its performance when the modulation mechanism is distorted with random noise. This can be achieved multiplying the γ_0 and β_0 respectively by new terms γ_{noise} and β_{noise}, drawn from a normal distribution. Comparing the first and last rows in Table 2 we observe a 7.97% drop in test accuracy when distorting all the cross-modulation blocks, which indicates that the model has learned

[1] Our re-implementation.

[2] We confirmed this through personal communication with the paper's authors.



(a) γ_0 and β_0 distributions

(b) Analysis of weight matrix W

(c) Averaged norms per block

Figure 2: Block-level modulation analysis: (a) distribution of absolute values of γ_0 and β_0 post-multipliers, (b) decomposition in self-modulation and cross modulation termsanalysis, and (c) Average norms per block.

to take advantage of them during training. We employed $\mathcal{N}(1, 0.3)$ for all the experiments in Table 2, but similar conclusions are reached from other standard deviation values.

The model applies cross-modulation at different levels of abstraction We assess whether the model learns to take advantage of cross-modulation at various levels of abstraction in the network in accordance to our motivating hypothesis. We perform an ablation study similar to the one above where we randomly distort the modulation selectively in different blocks (Table 2, rows 2-4). Our results suggest that the model has learned to take advantage of cross-modulation at multiple levels of abstraction, but relies more heavily on modulations in the second and fourth blocks. These results match the absolute value distribution observed for the γ_0 and β_0 post-multipliers (Figure 2), which regulate the intensity of the modulation on a per channel basis. We see that the γ_0 distribution is slightly higher for blocks 2 and 4.

Note that in this analysis we do not *train* the model with fewer modulated blocks; in that case, it is possible that it could learn to adapt to a different architectural configuration. More experiments are needed to quantify the effect of the specific number and location of modulated blocks on overall performance.

The model cross-modulates *and* self-modulates Recent work such as [3] outlines the benefit of *self-modulation* — where the feature extraction pipeline interacts with itself — in a large-scale classification setting. The noncommutative nature of our proposed cross-modulation mechanism allows both *self-modulation* and *cross-modulation*. To disambiguate between the two, we analyze the weight matrices of the FiLM generator blocks. As Figure 2b shows, these weight matrices can be viewed as the concatenation of two submatrices responsible for self-modulation and cross-modulation, respectively. Figure 2c shows the average weight matrix column norm for the self- and cross-modulation submatrices at each block. We see that self-modulation has a greater influence on the predicted FiLM parameters, but that the model also takes advantage of cross-modulation.

4 Conclusion

We have proposed a new architectural feature called *cross-modulation* which allows support and query examples to interact at multiple levels of abstraction and which can be used in conjunction with metric learning approaches to few-shot classification. Initial experiments with a baseline Matching Networks architecture show promising results in the *5-way, 1-shot* setting, and our analysis of the trained network shows that the model equipped with cross-modulation learns to use it in meaningful ways. Interesting avenues for future work include more thorough empirical verification (e.g. on more datasets and network architectures), developing analogous cross-modulation architectures for other methods such as Prototypical Networks, and addressing scaling issues associated with the use of pairwise interactions at multiple levels of abstraction.

References

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2014.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CVPR's ILSVRC 2017 Workshop*, 2017.

[4] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, 2016.

[5] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *EMNLP*, 2018.

[6] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *SSW*, 2016.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[8] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR*, 2018.

[9] Taesup Kim, Jaesik Yoon, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. *arXiv preprint arXiv:1806.03836*, 2018.

[10] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *CoRR*, 2018.

[11] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *CoRR*, 2018.

[12] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *CoRR*, 2018.

[13] Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29 (NIPS)*, 2016.

[14] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30 (NIPS)*, 2017.

[15] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[16] Boris N. Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems 31*, 2018.

[17] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[18] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016.

[19] Tsendsuren Munkhdalai, Xingdi Yuan, Soroush Mehri, and Adam Trischler. Rapid adaptation with conditionally shifted neurons. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[20] Pranav Shyam, Shubham Gupta, and Ambedkar Dukkipati. Attentive recurrent comparators. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[21] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations (ICLR)*, 2018.

[22] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations (ICLR)*, 2018.

[23] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015.

[24] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *AAAI*. AAAI Press, 2018.

[25] Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014.

[26] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *CoRR*, 2017.