Feature-wise transformations

A simple and surprisingly effective family of conditioning mechanisms.

AUTHORS

Vincent Dumoulin
Ethan Perez
Nathan Schucher
Florian Strub
Harm de Vries
Aaron Courville
Yoshua Bengio

AFFILIATIONS

Google Brain
Rice University, MILA
Element AI
Univ. of Lille, Inria
MILA
MILA
MILA

PUBLISHED

July 9, 2018

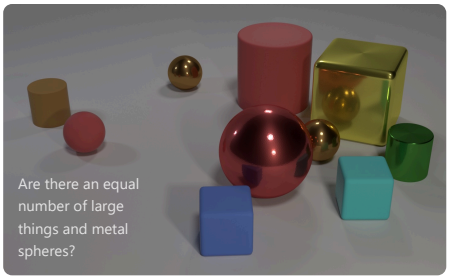
DOI

10.23915/distill.00011

Many real-world problems require integrating multiple sources of information. Sometimes these problems involve multiple, distinct modalities of information — vision, language, audio, etc. — as is required to understand a scene in a movie or answer a question about an image. Other times, these problems involve multiple sources of the same kind of input, i.e. when summarizing several documents or drawing one image in the style of another.

Video and audio must be understood in the context of each other to understand the scene.

Credit: still frame from the movie Charade.



An image needs to be processed in the context of a question being asked.

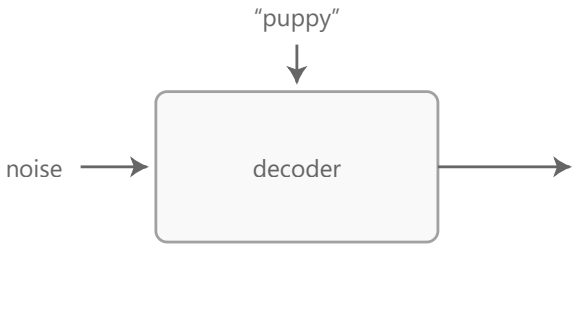
Credit: image-question pair from the CLEVR dataset.

When approaching such problems, it often makes sense to process one source of information *in the context of* another; for instance, in the right example above, one can extract meaning from the image in the context of the question. In machine learning, we often refer to this context-based processing as *conditioning*: the computation carried out by a model is conditioned or *modulated* by information extracted from an auxiliary input.

Finding an effective way to condition on or fuse sources of information is an open research problem, and in this article, we concentrate on a specific family of approaches we call *feature-wise transformations*. We will examine the use of feature-wise transformations in many neural network architectures to solve a surprisingly large and diverse set of problems; their success, we will argue, is due to being flexible enough to learn an effective representation of the conditioning input in varied settings. In the language of multi-task learning, where the conditioning signal is taken to be a task description, feature-wise transformations learn a task representation which allows them to capture and leverage the relationship between multiple sources of information, even in remarkably different problem settings.

Feature-wise transformations

To motivate feature-wise transformations, we start with a basic example, where the two inputs are images and category labels, respectively. For the purpose of this example, we are interested in building a generative model of images of various classes (puppy, boat, airplane, etc.). The model takes as input a class and a source of random noise (e.g., a vector sampled from a normal distribution) and outputs an image sample for the requested class.



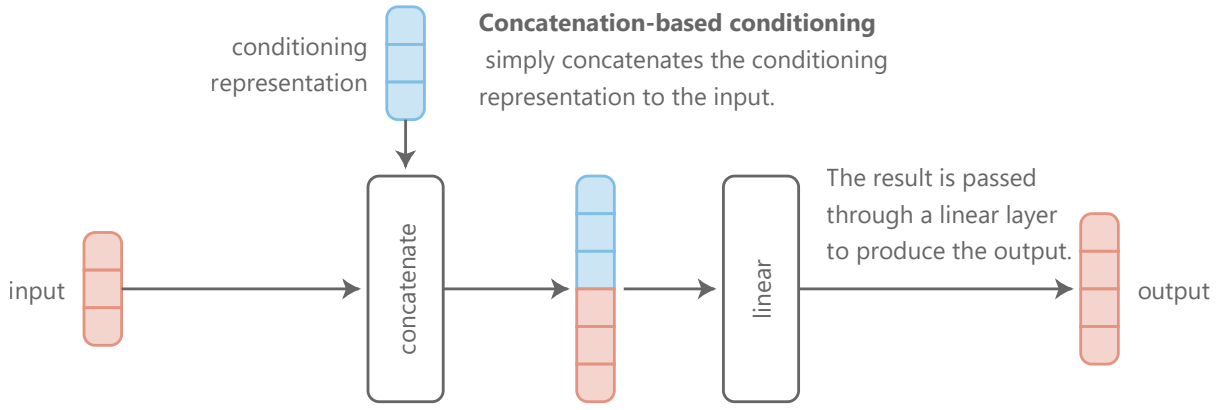
A **decoder-based generative model** maps a source of noise to a sample in the context of the "puppy" class.

Our first instinct might be to build a separate model for each class. For a small number of classes this approach is not too bad a solution, but for thousands of classes, we quickly run into scaling issues, as the number of parameters to store and train grows with the number of classes. We are also missing out on the opportunity to leverage commonalities between classes; for instance, different types of dogs (puppy, terrier, dalmatian, etc.) share visual traits and are likely to share computation when mapping from the abstract noise vector to the output image.

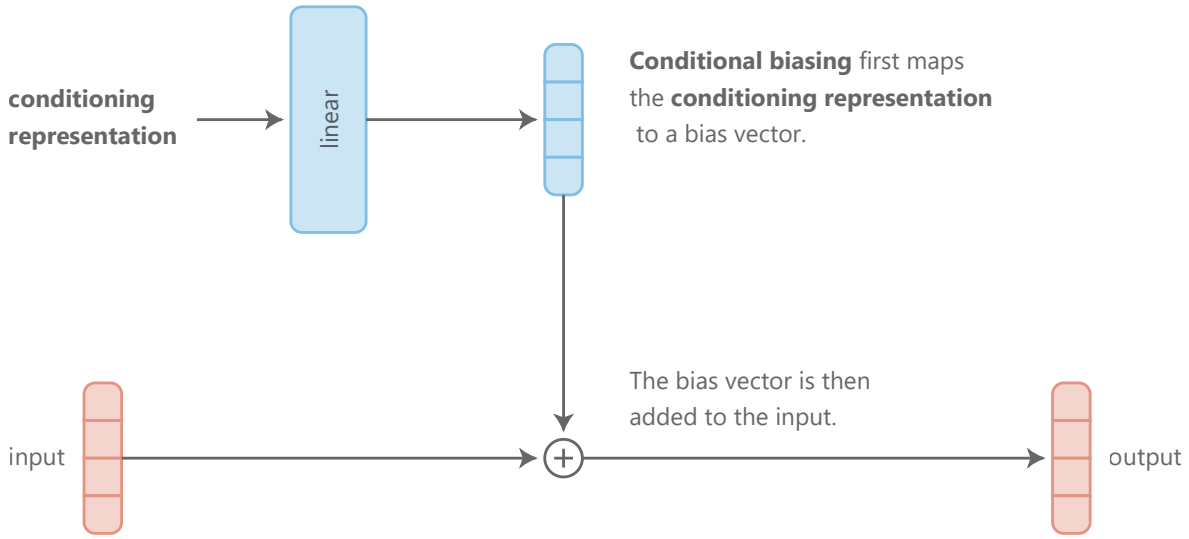
Now let's imagine that, in addition to the various classes, we also need to model attributes like size or color. In this case, we can't reasonably expect to train a separate network for *each* possible conditioning combination! Let's examine a few simple options.

A quick fix would be to concatenate a representation of the conditioning information to the noise vector and treat the result as the model's input. This solution is quite parameter-efficient, as we only need to increase the size of the first layer's weight matrix. However, this approach makes the implicit assumption that the input is where the model needs to use the conditioning information. Maybe this assumption is correct, or maybe it's not; perhaps the model does not need to incorporate the conditioning information until late into the generation process (e.g., right before generating the final pixel output when conditioning on texture). In this case, we would be forcing the model to carry this information around unaltered for many layers.

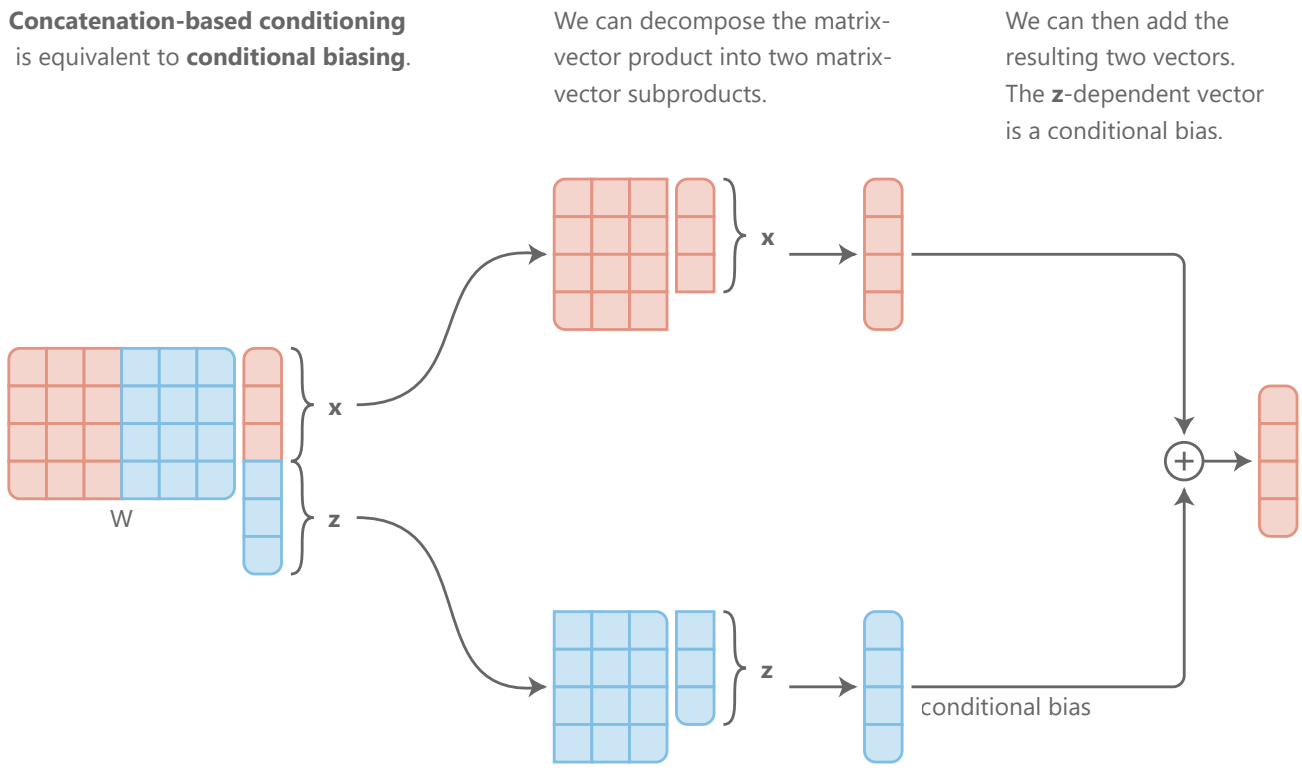
Because this operation is cheap, we might as well avoid making any such assumptions and concatenate the conditioning representation to the input of *all* layers in the network. Let's call this approach *concatenation-based conditioning*.



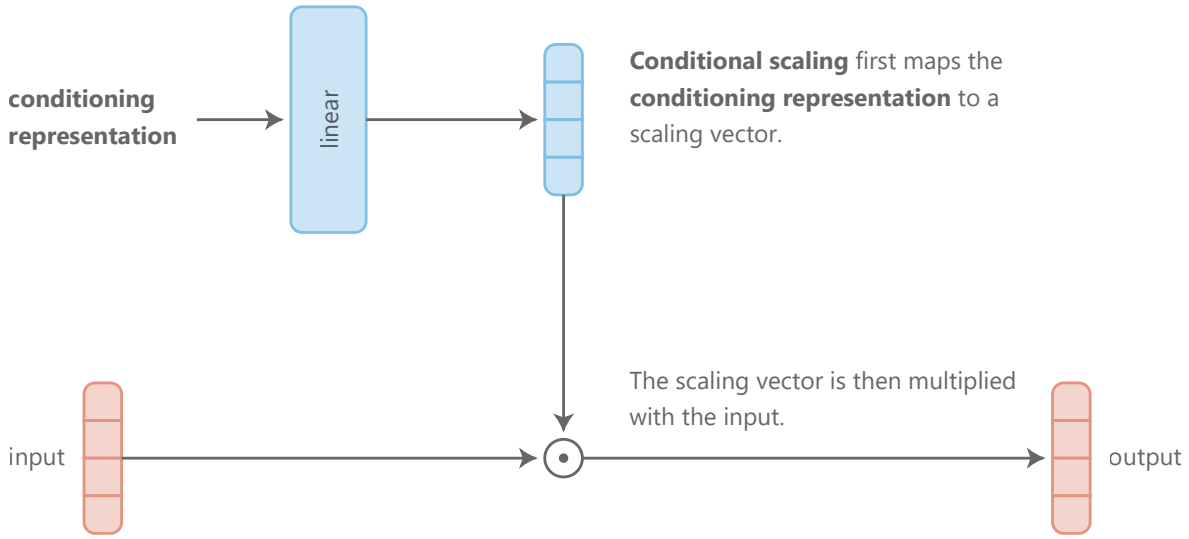
Another efficient way to integrate conditioning information into the network is via *conditional biasing*, namely, by adding a *bias* to the hidden layers based on the conditioning representation.



Interestingly, conditional biasing can be thought of as another way to implement concatenation-based conditioning. Consider a fully-connected linear layer applied to the concatenation of an input $\mathbf{x}$ and a conditioning representation $\mathbf{z}$: [1]



Yet another efficient way to integrate class information into the network is via *conditional scaling*, i.e., scaling hidden layers based on the conditioning representation.



A special instance of conditional scaling is feature-wise sigmoidal gating: we scale each feature by a value between 0 and 1 (enforced by applying the logistic function), as a function of the conditioning representation. Intuitively, this gating allows the conditioning information to select which features are passed forward and which are zeroed out.

Given that both additive and multiplicative interactions seem natural and intuitive, which approach should we pick? One argument in favor of *multiplicative* interactions is that they are useful in learning relationships between inputs, as these interactions naturally identify "matches": multiplying elements that agree in sign yields larger values than multiplying elements that disagree. This property is why dot products are often used to determine how similar two vectors are. [2] One argument in favor of *additive* interactions is that they are more natural for applications that are less strongly dependent on the joint values of two inputs, like feature aggregation or feature detection (i.e., checking if a feature is present in either of two inputs).

In the spirit of making as few assumptions about the problem as possible, we may as well combine *both* into a conditional *affine transformation*. [3]

All methods outlined above share the common trait that they act at the *feature* level; in other words, they leverage *feature-wise* interactions between the conditioning representation and the conditioned network. It is certainly possible to use more complex interactions, but feature-wise interactions often strike a happy compromise between effectiveness and efficiency: the number of scaling and/or shifting coefficients to predict scales linearly with the number of features in the network. Also, in practice, feature-wise transformations (often compounded across multiple layers) frequently have enough capacity to model complex phenomenon in various settings.

Lastly, these transformations only enforce a limited inductive bias and remain domain-agnostic. This quality can be a downside, as some problems may be easier to solve with a stronger inductive bias. However, it is this characteristic which also enables these transformations to be so widely effective across problem domains, as we will later review.

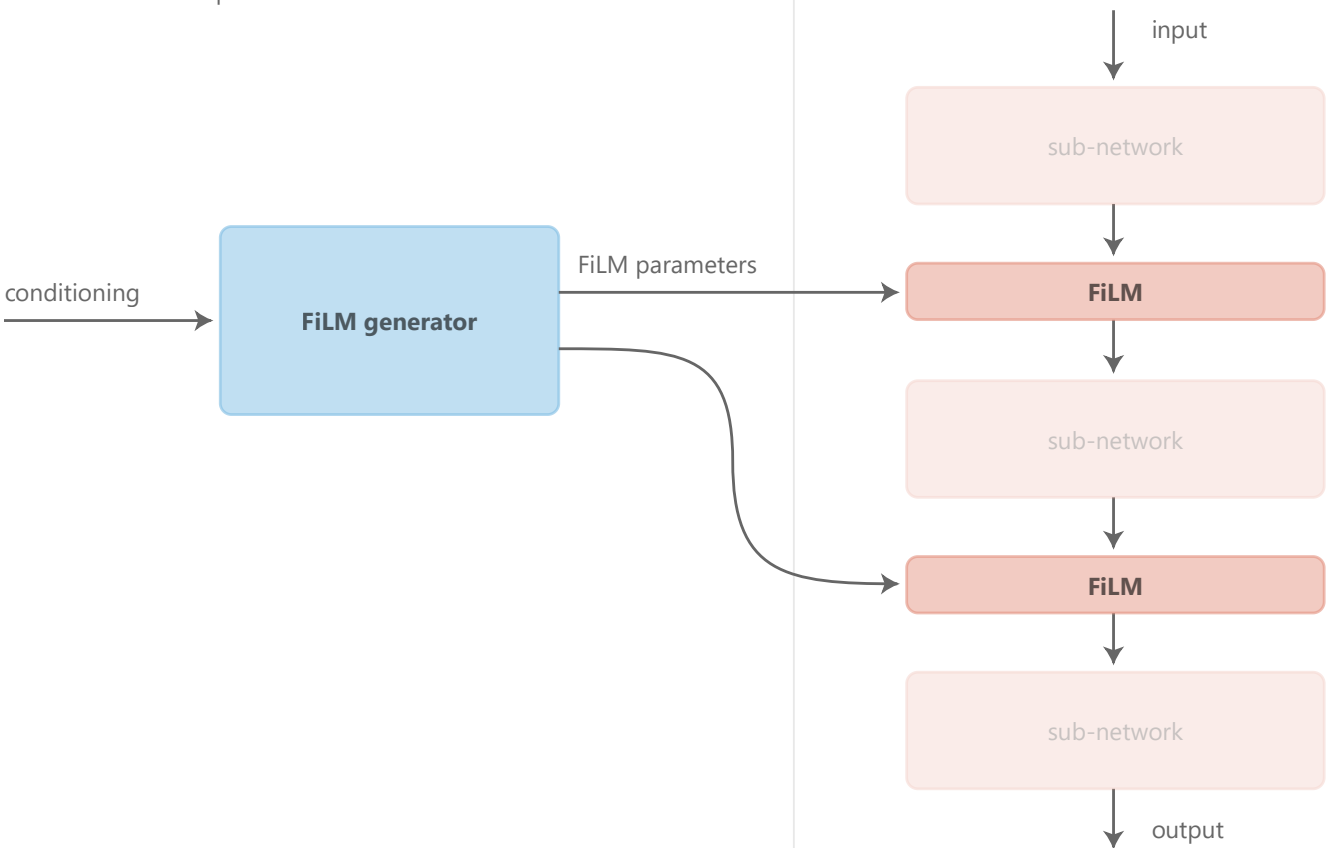
Nomenclature

To continue the discussion on feature-wise transformations we need to abstract away the distinction between multiplicative and additive interactions. Without losing generality, let's focus on feature-wise affine transformations, and let's adopt the nomenclature of Perez et al. [1], which formalizes conditional affine transformations under the acronym *FiLM*, for Feature-wise Linear Modulation. [4]

We say that a neural network is modulated using FiLM, or *FiLM-ed*, after inserting *FiLM layers* into its architecture. These layers are parametrized by some form of conditioning information, and the mapping from conditioning information to FiLM parameters (i.e., the shifting and scaling coefficients) is called the *FiLM generator*. In other words, the FiLM generator predicts the parameters of the FiLM layers based on some auxiliary input. Note that the FiLM parameters are parameters in one network but predictions from another network, so they aren't learnable parameters with fixed weights as in the fully traditional sense. For simplicity, you can assume that the FiLM generator outputs the concatenation of all FiLM parameters for the network architecture.

The **FiLM generator** processes the conditioning information and produces parameters that describe how the target network should alter its computation.

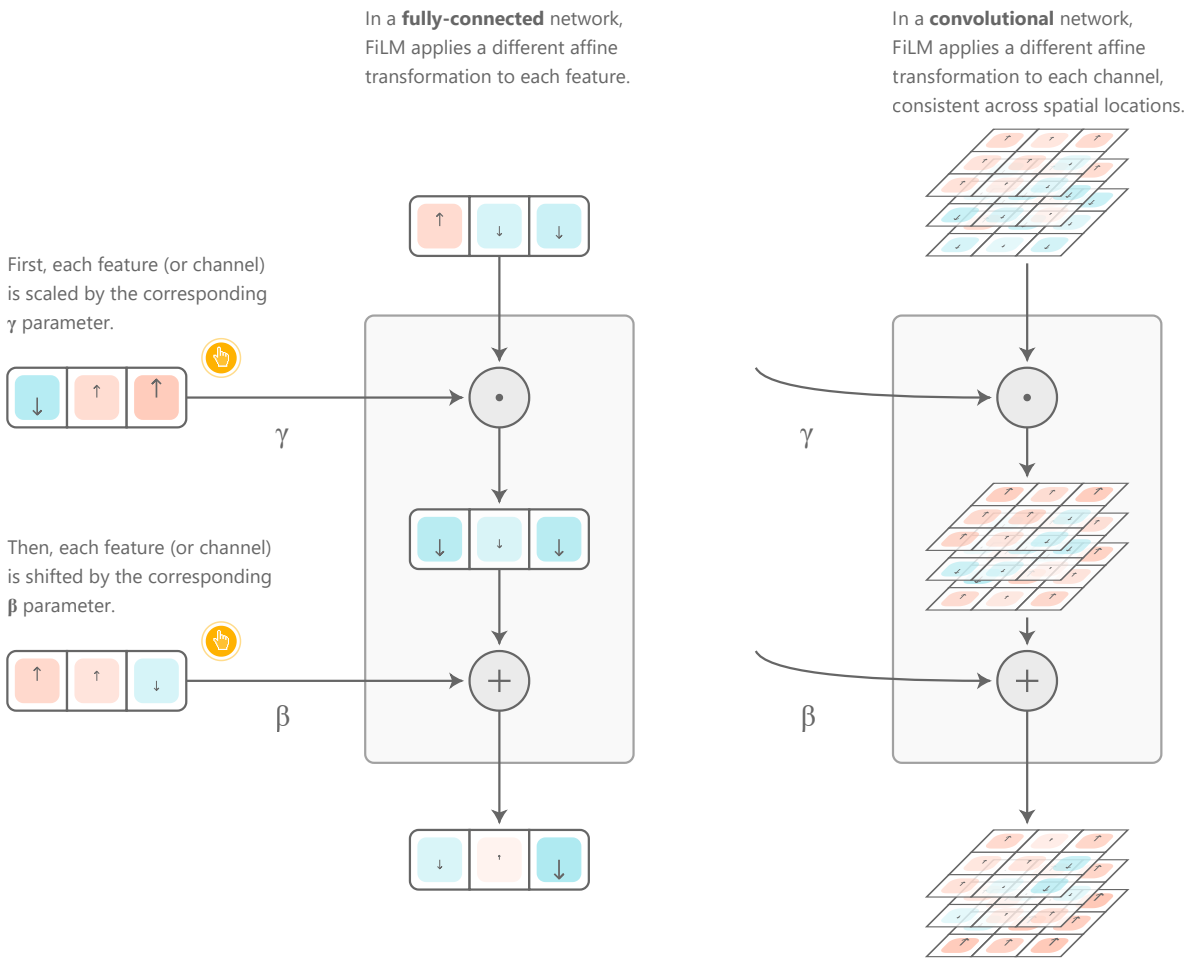
Here, the **FiLM-ed network**'s computation is conditioned by two FiLM layers.



As the name implies, a FiLM layer applies a feature-wise affine transformation to its input. By *feature-wise*, we mean that scaling and shifting are applied element-wise, or in the case of convolutional networks, feature map -wise. [5] In other words, assuming $\mathbf{x}$ is a FiLM layer's input, $\mathbf{z}$ is a conditioning input, and γ and β are $\mathbf{z}$-dependent scaling and shifting vectors,

$$\text{FiLM}(\mathbf{x}) = \gamma(\mathbf{z}) \odot \mathbf{x} + \beta(\mathbf{z}).$$

You can interact with the following fully-connected and convolutional FiLM layers to get an intuition of the sort of modulation they allow:



In addition to being a good abstraction of conditional feature-wise transformations, the FiLM nomenclature lends itself well to the notion of a *task representation*. From the perspective of multi-task learning, we can view the conditioning signal as the task description. More specifically, we can view the concatenation of all FiLM scaling and shifting coefficients as both an instruction on *how to modulate* the conditioned network and a *representation* of the task at hand. We will explore and illustrate this idea later on.

Feature-wise transformations in the literature

Feature-wise transformations find their way into methods applied to many problem settings, but because of their simplicity, their effectiveness is seldom highlighted in lieu of other novel research contributions. Below are a few notable examples of feature-wise transformations in the literature, grouped by application domain. The diversity of these applications underscores the flexible, general-purpose ability of feature-wise interactions to learn effective task representations.

expand all

Visual question-answering +

Style transfer +

Image recognition +

Natural language processing +

Reinforcement learning +

Generative modeling +

Speech recognition +

Domain adaptation and few-shot learning +

Related ideas

Aside from methods which make direct use of feature-wise transformations, the FiLM framework connects more broadly with the following methods and concepts.

expand all

Zero-shot learning +

HyperNetworks +

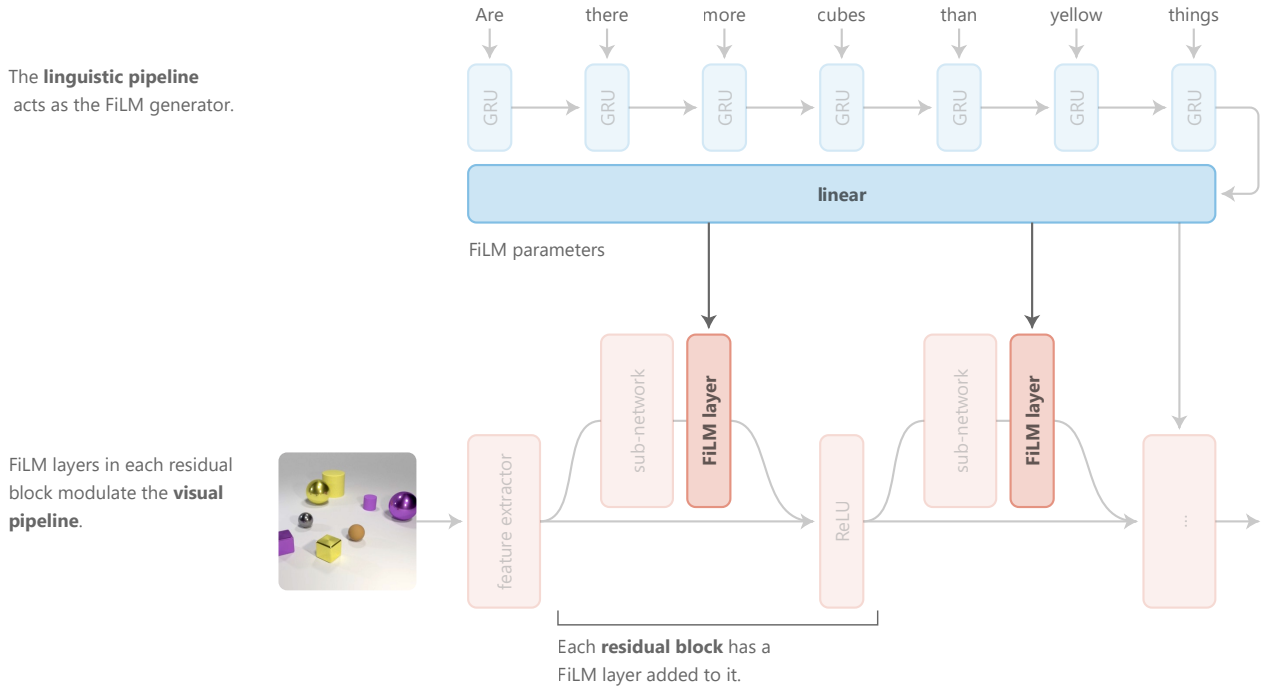
Attention +

Bilinear transformations +

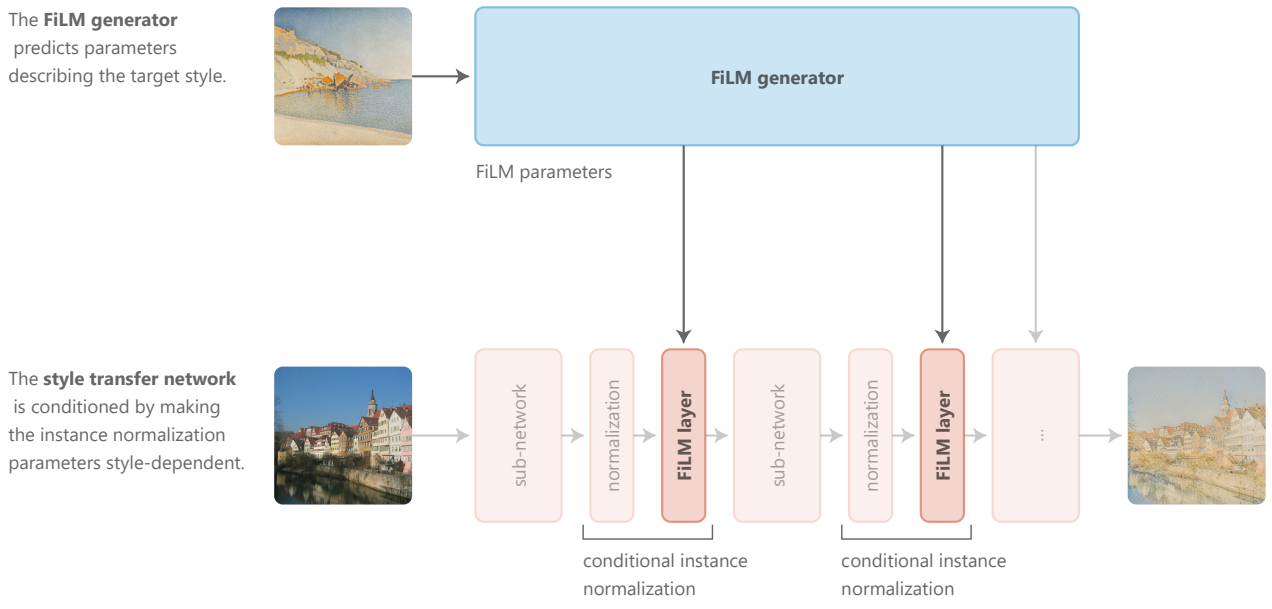
Properties of the learned task representation

As hinted earlier, in adopting the FiLM perspective we implicitly introduce a notion of *task representation*: each task — be it a question about an image or a painting style to imitate — elicits a different set of FiLM parameters via the FiLM generator which can be understood as its representation in terms of how to modulate the FiLM-ed network. To help better understand the properties of this representation, let's focus on two FiLM-ed models used in fairly different problem settings:

- The visual reasoning model of Perez et al. [2, 1], which uses FiLM to modulate a visual processing pipeline based off an input question.



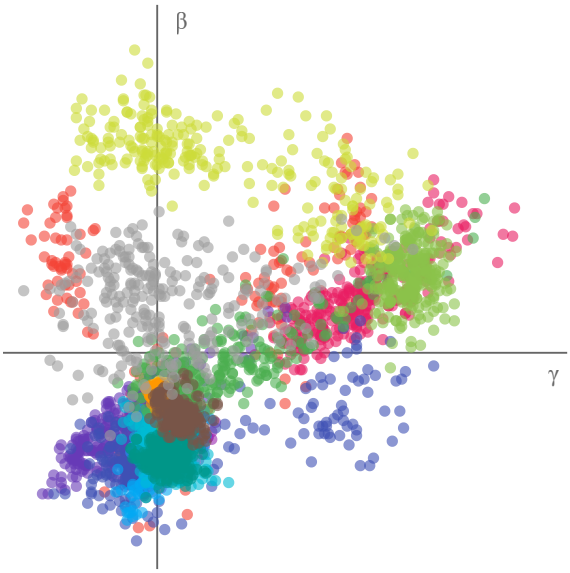
- The artistic style transfer model of Ghiasi et al. [10], which uses FiLM to modulate a feed-forward style transfer network based off an input style image.



As a starting point, can we discern any pattern in the FiLM parameters as a function of the task description? One way to visualize the FiLM parameter space is to plot γ against β, with each point corresponding to a specific task description and a specific feature map. If we color-code each point according to the feature map it belongs to we observe the following:

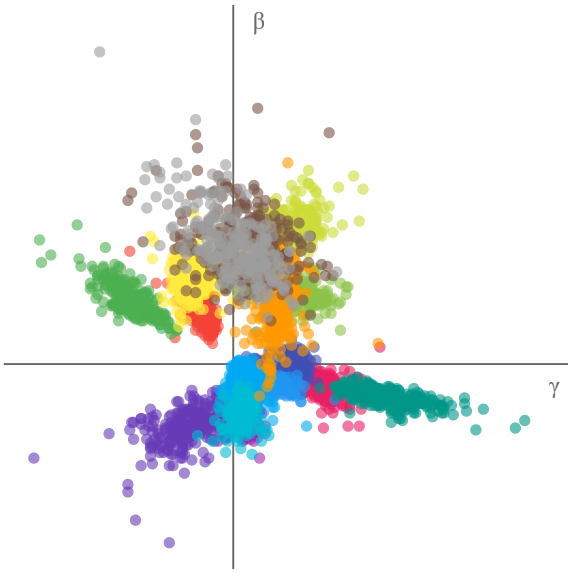
FiLM parameters for **256 tasks** and for **16 feature maps**, chosen randomly.

Visual reasoning model



Feature map

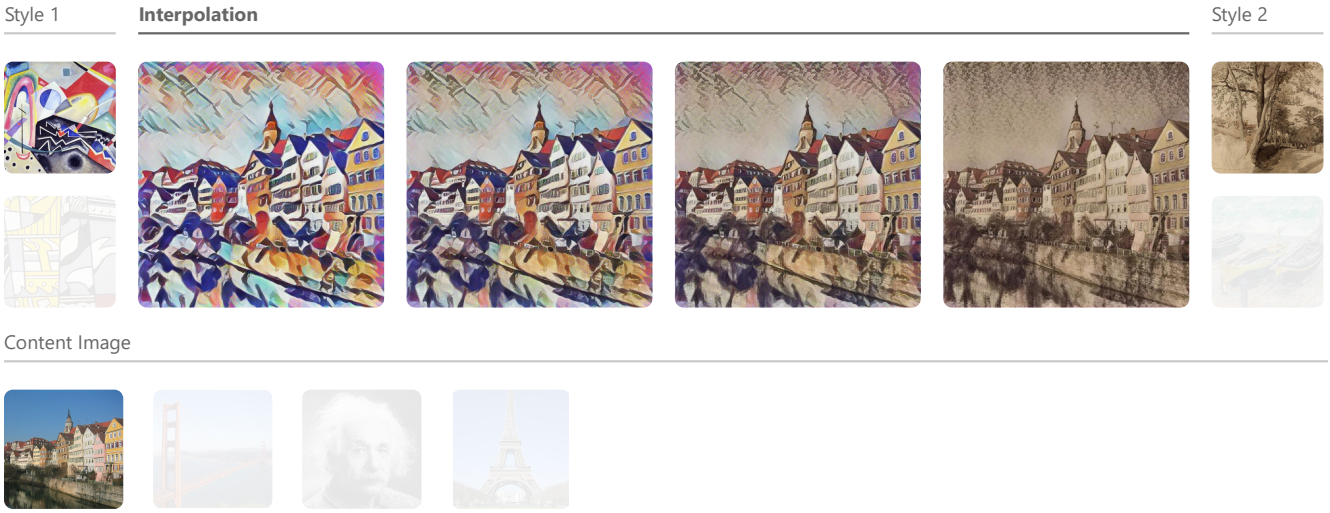
Style transfer model



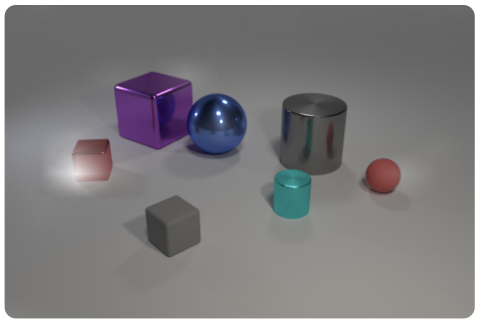
Feature map

The plots above allow us to make several interesting observations. First, FiLM parameters cluster by feature map in parameter space, and the cluster locations are not uniform across feature maps. The orientation of these clusters is also not uniform across feature maps: the main axis of variation can be γ-aligned, β-aligned, or diagonal at varying angles. These findings suggest that the affine transformation in FiLM layers is not modulated in a single, consistent way, i.e., using γ only, β only, or γ and β together in some specific way. Maybe this is due to the affine transformation being overspecified, or maybe this shows that FiLM layers can be used to perform modulation operations in several distinct ways.

Nevertheless, the fact that these parameter clusters are often somewhat "dense" may help explain why the style transfer model of Ghiasi et al. [10] is able to perform style interpolations: any convex combination of FiLM parameters is likely to correspond to a meaningful parametrization of the FiLM-ed network.

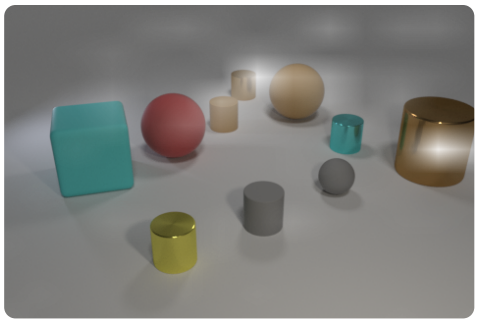


To some extent, the notion of interpolating between tasks using FiLM parameters can be applied even in the visual question-answering setting. Using the model trained in Perez et al. [1], we interpolated between the model's FiLM parameters for two pairs of CLEVR questions. Here we visualize the input locations responsible for the globally max-pooled features fed to the visual pipeline's output classifier:



What shape is the red thing left of the sphere?

What shape is the red thing right of the sphere?



How many brown things are there?

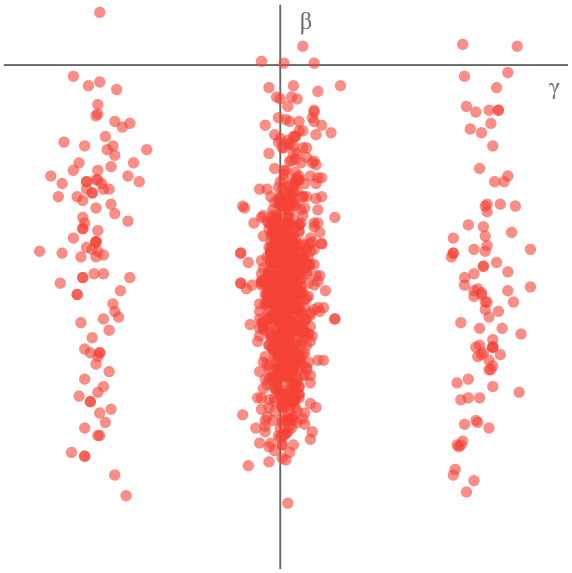
How many yellow things are there?

The network seems to be softly switching where in the image it is looking, based on the task description. It is quite interesting that these semantically meaningful interpolation behaviors emerge, as the network has not been trained to act this way.

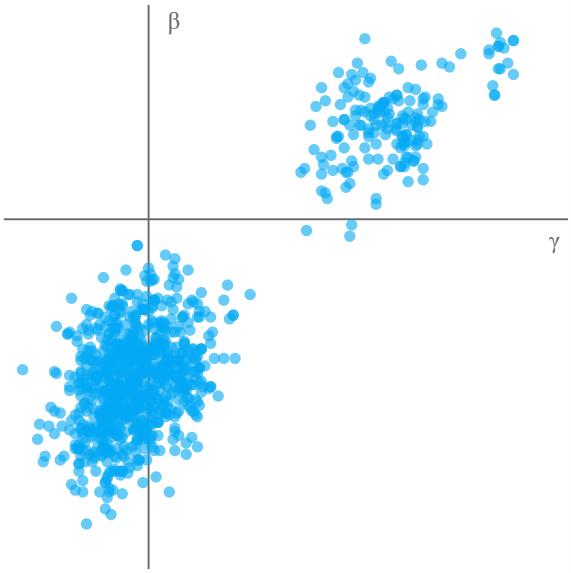
Despite these similarities across problem settings, we also observe qualitative differences in the way in which FiLM parameters cluster as a function of the task description. Unlike the style transfer model, the visual reasoning model sometimes exhibits several FiLM parameter sub-clusters for a given feature map.

FiLM parameters of the **visual reasoning model** for 256 questions chosen randomly.

Feature map 26 of the first FiLM layer.



Feature map 76 of the first FiLM layer.

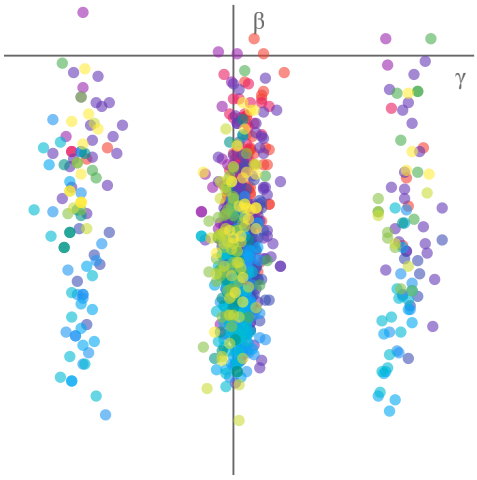


At the very least, this may indicate that FiLM learns to operate in ways that are problem-specific, and that we should not expect to find a unified and problem-independent explanation for FiLM's success in modulating FiLM-ed networks. Perhaps the compositional or discrete nature of visual reasoning requires the model to implement several well-defined modes of operation which are less necessary for style transfer.

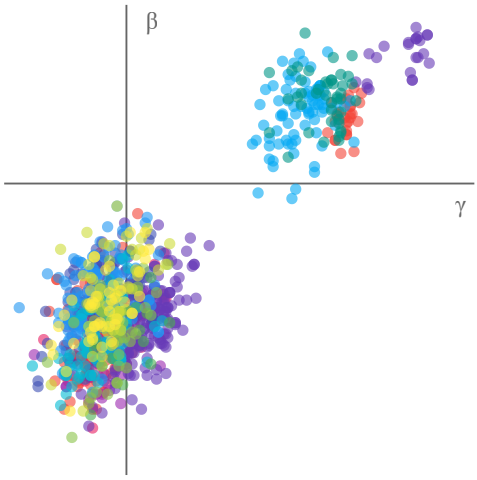
Focusing on individual feature maps which exhibit sub-clusters, we can try to infer how questions regroup by color-coding the scatter plots by question type.

FiLM parameters of the **visual reasoning model** for 256 questions chosen randomly.

Feature map 26 of the first FiLM layer.



Feature map 76 of the first FiLM layer.



Question type

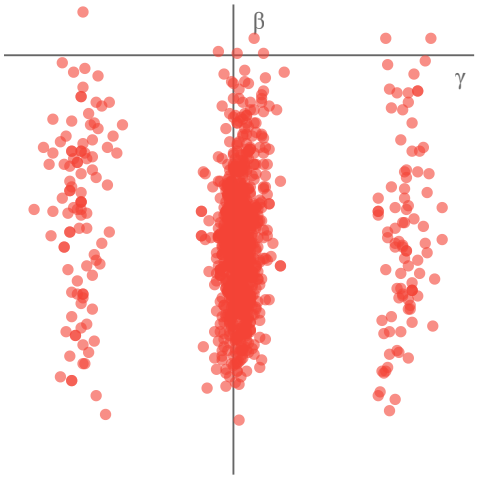
- Exists
- Less than
- Greater than
- Count
- Query material
- Query size
- Query color
- Query shape
- Equal color
- Equal integer
- Equal shape
- Equal size
- Equal material

Sometimes a clear pattern emerges, as in the right plot, where color-related questions concentrate in the top-right cluster — we observe that questions either are of type *Query color* or *Equal color*, or contain concepts related to color. Sometimes it is harder to draw a conclusion, as in the left plot, where question types are scattered across the three clusters.

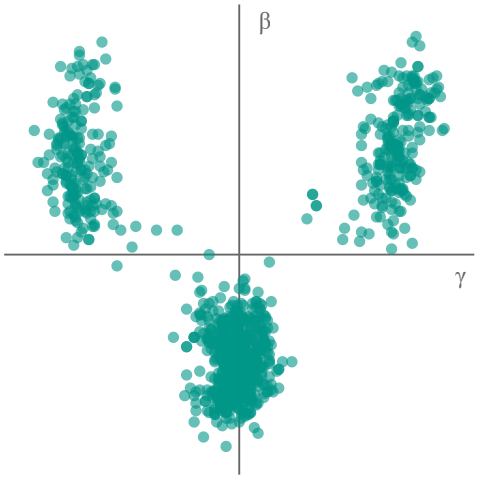
In cases where question types alone cannot explain the clustering of the FiLM parameters, we can turn to the conditioning content itself to gain an understanding of the mechanism at play. Let's take a look at two more plots: one for feature map 26 as in the previous figure, and another for a different feature map, also exhibiting several subclusters. This time we regroup points by the words which appear in their associated question.

FiLM parameters of the **visual reasoning model** for 256 questions chosen randomly.

Feature map 26 suggests an object position separation mechanism.



Feature map 92 suggests an object material separation mechanism.



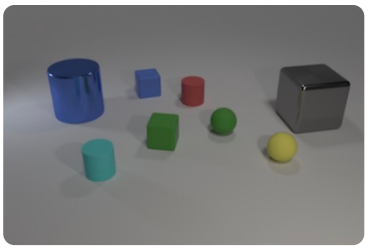
Word in question

front
behind
left
right
material
rubber
matte
metal
metallic
shiny

In the left plot, the left subcluster corresponds to questions involving objects positioned *in front* of other objects, while the right subcluster corresponds to questions involving objects positioned *behind* other objects. In the right plot we see some evidence of separation based on object material: the left subcluster corresponds to questions involving *matte* and *rubber* objects, while the right subcluster contains questions about *shiny* or *metallic* objects.

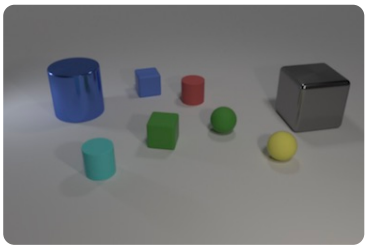
The presence of sub-clusters in the visual reasoning model also suggests that question interpolations may not always work reliably, but these sub-clusters don't preclude one from performing arithmetic on the question representations, as Perez et al. [1] report.

The model incorrectly answers a question which involves an unseen combination of concepts (in **bold**).



*Q: What is the **blue** big **cylinder** made of?*

Rather than using the FiLM parameters of the FiLM generator, we can use those produced by combining questions with familiar combinations of concepts (in **bold**). This corrects the model's answer.



*Q_A : What is the **blue** big **sphere** made of?*

*Q_B : What is the **green** big **cylinder** made of?*

*Q_C : What is the **green** big **sphere** made of?*

Perez et al. [1] report that this sort of task analogy is not always successful in correcting the model's answer, but it does point to an interesting fact about FiLM-ed networks: sometimes the model makes a mistake not because it is incapable of computing the correct output, but because it fails to produce the correct FiLM parameters for a given task description. The reverse can also be true: if the set of tasks the model was trained on is insufficiently rich, the computational primitives learned by the FiLM-ed network may be insufficient to ensure good generalization. For instance, a style transfer model may lack the ability to produce zebra-like patterns if there are no stripes in the styles it was trained on. This could explain why Ghiasi et al. [10] report that their style transfer model's ability to produce pastiches for new styles degrades if it has been trained on an insufficiently large number of styles. Note however that in that case the FiLM generator's failure to generalize could also play a role, and further analysis would be needed to draw a definitive conclusion.

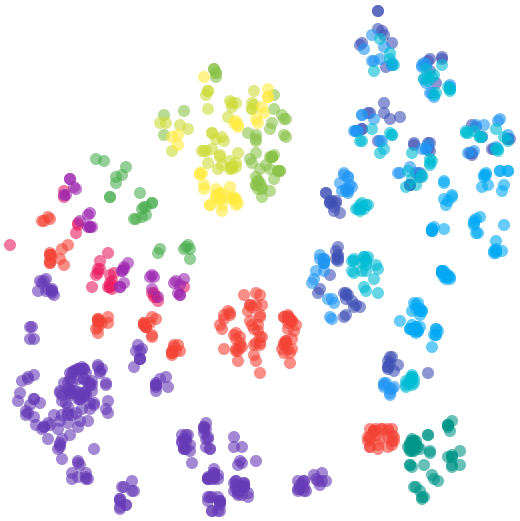
This points to a separation between the various computational primitives learned by the FiLM-ed network and the "numerical recipes" learned by the FiLM generator: the model's ability to generalize depends both on its ability to parse new forms of task descriptions and on it having learned the required computational primitives to solve those tasks. We note that this multi-faceted notion of generalization is inherited directly from the multi-task point of view adopted by the FiLM framework.

Let's now turn our attention back to the overal structural properties of FiLM parameters observed thus far. The existence of this structure has already been explored, albeit more indirectly, by Ghiasi et al. [10] as well as Perez et al. [1], who applied t-SNE [38] on the FiLM parameter values.

t-SNE projection of FiLM parameters for many task descriptions.

Visual reasoning model

Question type



- Exists
- Less than
- Greater than
- Count
- Query material
- Query size
- Query color
- Query shape
- Equal color
- Equal integer
- Equal shape
- Equal size
- Equal material

Reset pan / zoom

Style transfer model

Artist name

Reset pan / zoom

The projection on the left is inspired by a similar projection done by Perez et al. [1] for their visual reasoning model trained on CLEVR and shows how questions group by question type. The projection on the right is inspired by a similar projection done by Ghiasi et al. [10] for their style transfer network. The projection does not cluster artists as neatly as the projection on the left, but this is to be expected, given that an artist's style may vary widely over time. However, we can still detect interesting patterns in the projection: note for instance the isolated cluster (circled in the figure) in which paintings by Ivan Shishkin and Rembrandt are aggregated. While these two painters exhibit fairly different styles, the cluster is a grouping of their sketches.

Rembrandt's *Woman with a Pink*.

Shishkin's *Woman with a boy in the forest*.

Sketches by Rembrandt and Shishkin found in the same t-SNE cluster.

To summarize, the way neural networks learn to use FiLM layers seems to vary from problem to problem, input to input, and even from feature to feature; there does not seem to be a single mechanism by which the network uses FiLM to condition computation. This flexibility may explain why FiLM-related methods have been successful across such a wide variety of domains.

Discussion

Looking forward, there are still many unanswered questions. Do these experimental observations on FiLM-based architectures generalize to other related conditioning mechanisms, such as conditional biasing, sigmoidal gating, HyperNetworks, and bilinear transformations? When do feature-wise transformations outperform methods with stronger inductive biases and vice versa? Recent work combines feature-wise

transformations with stronger inductive bias methods [4, 22, 39], which could be an optimal middle ground. Also, to what extent are FiLM's task representation properties inherent to FiLM, and to what extent do they emerge from other features of neural networks (i.e. non-linearities, FiLM generator depth, etc.)? If you are interested in exploring these or other questions about FiLM, we recommend looking into the code bases for FiLM models for visual reasoning [1] and style transfer [10] which we used as a starting point for our experiments here.

Finally, the fact that changes on the feature level alone are able to compound into large and meaningful modulations of the FiLM-ed network is still very surprising to us, and hopefully future work will uncover deeper explanations. For now, though, it is a question that evokes the even grander mystery of how neural networks in general compound simple operations like matrix multiplications and element-wise non-linearities into semantically meaningful transformations.

Bibliographic Notes

Multiplicative interactions have succeeded on various tasks, ever since they were introduced in vision as "mapping units" [40] and "dynamic mappings" [41] around 40 years ago. These tasks include Character-level Language Modeling [42], Image Denoising [43], Pose Estimation [44], Tracking [45, 46], Action Recognition [47, 48], and, more generally, tasks involving relating or matching inputs, such as from different modalities or points in time [49].

Many models lie on the spectrum between FiLM and Hypernetworks:

- Adaptive CNN [50] predicts the value of several of the model's convolution filters as a function of auxiliary inputs like camera perspective, level of noise, etc. The resulting convolution filters turn out to be very effective in difficult vision tasks such as crowd counting or image deblurring.
- Residual Adapters [51] also propose to predict entire convolutional filters conditioned on the visual recognition domain they are operating in.
- In zero-shot/one-shot learning, Ba et al. [52] propose a model that predicts convolutional filters and classifiers weights based on textual descriptions of object classes.
- In reinforcement learning, Oh et al. [53] propose a model that computes the parameters of a convolutional policy network conditioned on the task description.

Tenenbaum and Freeman [54] first introduced bilinear models in the vision community to better disentangle latent perceptual factors. The authors wanted to separate an image's style from its content, arguing that classic linear models were not rich enough to extract such complex interaction. They demonstrate the effectiveness of their approach by applying it to spoken vowel identification or zero-shot font classification. Notable applications include:

- Chuang et al. [55] perform facial animation using bilinear transformations by separating key facial features (the style) from visual emotions (the content). Their method can modify a speaking subject's expression in recorded sequence from happy to angry or neutral.

- Chu and Park [56] and Yang et al. [57] apply bilinear models to recommendation systems by extracting user and item information in various settings. More generally, recommendation systems rely heavily on matrix factorization methods [58], which can be viewed as a bilinear model where one of the latent vectors is fixed [54].
- More recently, bilinear models have inspired new neural architectures in visual recognition [59], video action recognition [60], and visual question-answering [61].

Acknowledgements

This article would be nowhere near where it is today without the honest and constructive feedback we received from various people across several organizations. We would like to thank Chris Olah and Shan Carter from the Distill editorial team as well as Ludwig Schubert from the Google Brain team for being so generous with their time and advice. We would also like to thank Archy de Berker, Xavier Snelgrove, Pedro Oliveira Pinheiro, Alexei Nordell-Markovits, Masha Krol, and Minh Dao from Element AI; Roland Memisevic from TwentyBN; Dzmitry Bahdanau from MILA; Ameesh Shah and Will Levine from Rice University; Dhanush Radhakrishnan from Roivant Sciences; Raymond Cano from Plaid; Eleni Triantafillou from Toronto University; Olivier Pietquin and Jon Shlens from Google Brain; and Jérémie Mary from Criteo.

Discussion and Review

Review 1 - Anonymous
Review 2 - Anonymous
Review 3 - Chris Olah

Footnotes

1. The same argument applies to convolutional networks, provided we ignore the border effects due to zero-padding. [↩]
2. Multiplicative interactions alone have had a history of success in various domains — see Bibliographic Notes. [↩]
3. An affine transformation is a transformation of the form $y = m * x + b$. [↩]
4. Strictly speaking, *linear* is a misnomer, as we allow biasing, but we hope the more rigorous-minded reader will forgive us for the sake of a better-sounding acronym. [↩]
5. To expand a little more on the convolutional case, feature maps can be thought of as the same feature detector being evaluated at different spatial locations, in which case it makes sense to apply the same affine transformation to all spatial locations. [↩]
6. The authors also describe a location-dependent biasing scheme which cannot be expressed in terms of FiLM layers due to the absence of the feature-wise property. [↩]
7. As is commonly done to turn a linear transformation into an affine transformation. [↩]

References

1. **FiLM: Visual Reasoning with a General Conditioning Layer** [PDF]
Perez, E., Strub, F., Vries, H.d., Dumoulin, V. and Courville, A., 2018. AAAI.
2. **Learning visual reasoning without strong priors** [PDF]
Perez, E., de Vries, H., Strub, F., Dumoulin, V. and Courville, A., 2017. ICML Workshop on Machine Learning in Speech and Language Processing.
3. **CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning** [PDF]
Johnson, J., Li, F., Hariharan, B., Zitnick, L.C., van der Maaten, L. and Girshick, R., 2017. Proceedings of the Conference on Computer

Vision and Pattern Recognition.

4. **Visual Reasoning with Multi-hop Feature Modulation**
   Strub, F., Seurin, M., Perez, E., de Vries, H., Mary, J., Preux, P., Courville, A. and Pietquin, O., 2018. ECCV.

5. **GuessWhat?! Visual object discovery through multi-modal dialogue** [PDF]
   de Vries, H., Strub, F., Chandar, S., Pietquin, O., Larochelle, H. and Courville, A., 2017. Proceedings of the Conference on Computer Vision and Pattern Recognition.

6. **ReferItGame: Referring to objects in photographs of natural scenes** [PDF]
   Kazemzadeh, S., Ordonez, V., Matten, M. and Berg, T., 2014. Conference on Empirical Methods in Natural Language Processing.

7. **Modulating early visual processing by language** [PDF]
   de Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O. and Courville, A., 2017. Advances in Neural Information Processing Systems.

8. **VQA: visual question answering** [PDF]
   Agrawal, A., Lu, J., Antol, S., Mitchell, M., Zitnick, C.L., Batra, D. and Parikh, D., 2015. Proceedings of the International Conference on Computer Vision.

9. **A learned representation for artistic style** [PDF]
   Dumoulin, V., Shlens, J. and Kudlur, M., 2017. Proceedings of the International Conference on Learning Representations.

10. **Exploring the structure of a real-time, arbitrary neural artistic stylization network** [PDF]
    Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V. and Shlens, J., 2017. Proceedings of the British Machine Vision Conference.

11. **Efficient video object segmentation via network modulation** [PDF]
    Yang, L., Wang, Y., Xiong, X., Yang, J. and Katsaggelos, A.K., 2018. arXiv.

12. **Arbitrary style transfer in real-time with adaptive instance normalization** [PDF]
    Huang, X. and Belongie, S., 2017. Proceedings of the International Conference on Computer Vision.

13. **Highway networks** [PDF]
    Srivastava, R.K., Greff, K. and Schmidhuber, J., 2015. ICML Deep Learning Workshop.

14. **Long short-term memory** [link]
    Hochreiter, S. and Schmidhuber, J., 1997. Neural Computation, Vol 9(8), pp. 1735--1780. MIT Press.

15. **Squeeze-and-Excitation networks** [PDF]
    Hu, J., Shen, L. and Sun, G., 2017. CVPR's ILSVRC 2017 Workshop.

16. **On the state of the art of evaluation in neural language models** [PDF]
    Melis, G., Dyer, C. and Blunsom, P., 2017. Proceedings of the International Conference on Learning Representations.

17. **Language modeling with gated convolutional networks** [PDF]
    Dauphin, Y.N., Fan, A., Auli, M. and Grangier, D., 2017. Proceedings of the International Conference on Machine Learning.

18. **Convolution sequence-to-sequence learning** [PDF]
    Gehring, J., Auli, M., Grangier, D., Yarats, D. and Dauphin, Y.N., 2017. Proceedings of the International Conference on Machine Learning.

19. **Gated-attention readers for text comprehension** [PDF]
    Dhingra, B., Liu, H., Yang, Z., Cohen, W.W. and Salakhutdinov, R., 2017. Proceedings of the Annual Meeting of the Association for Computational Linguistics.

20. **Gated-attention architectures for task-oriented language grounding** [PDF]
    Chaplot, D.S., Sathyendra, K.M., Pasumarthi, R.K., Rajagopal, D. and Salakhutdinov, R., 2017. ACL Workshop on Language Grounding for Robotics.

21. **Vizdoom: A doom-based AI research platform for visual reinforcement learning** [PDF]
Kempka, M., Wydmuch, M., Runc, G., Toczek, J. and Jaśkowski, W., 2016. IEEE Conference on Computational Intelligence and Games.

22. **Learning to follow language instructions with adversarial reward induction** [PDF]
Bahdanau, D., Hill, F., Leike, J., Hughes, E., Kohli, P. and Grefenstette, E., 2018. arXiv.

23. **Neural module networks** [PDF]
Andreas, J., Rohrbach, M., Darrell, T. and Klein, D., 2016. Proceedings of the Conference on Computer Vision and Pattern Recognition.

24. **Overcoming catastrophic forgetting in neural networks** [link]
Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D. and Hadsell, R., 2017. Proceedings of the National Academy of Sciences, Vol 114(13), pp. 3521--3526.

25. **Unsupervised representation learning with deep convolutional generative adversarial networks** [PDF]
Radford, A., Metz, L. and Chintala, S., 2016. Proceedings of the International Conference on Learning Representations.

26. **Generative adversarial nets** [PDF]
Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Advances in Neural Information Processing Systems.

27. **Conditional image generation with PixelCNN decoders** [PDF]
van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., Graves, A. and Kavukcuoglu, K., 2016. Advances in Neural Information Processing Systems.

28. **WaveNet: A generative model for raw audio** [PDF]
van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K., 2016. arXiv.

29. **Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition** [PDF]
Kim, T., Song, I. and Bengio, Y., 2017. Interspeech.

30. **Adaptive batch normalization for practical domain adaptation** [link]
Li, Y., Wang, N., Shi, J., Hou, X. and Liu, J., 2018. Pattern Recognition, Vol 80, pp. 109 - 117.

31. **TADAM: Task dependent adaptive metric for improved few-shot learning** [PDF]
Oreshkin, B.N., Rodriguez, P. and Lacoste, A., 2018. arXiv.

32. **Prototypical networks for few-shot learning** [PDF]
Snell, J., Swersky, K. and Zemel, R., 2017. Advances in Neural Information Processing Systems.

33. **Devise: A deep visual-semantic embedding model** [link]
Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J. and Mikolov, T., 2013. Advances in Neural Information Processing Systems, pp. 2121--2129.

34. **Zero-shot learning through cross-modal transfer** [PDF]
Socher, R., Ganjoo, M., Manning, C.D. and Ng, A., 2013. Advances in Neural Information Processing Systems, pp. 935--943.

35. **Zero-shot learning by convex combination of semantic embeddings** [PDF]
Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., Corrado, G.S. and Dean, J., 2014. Proceedings of the International Conference on Learning Representations.

36. **HyperNetworks** [PDF]
Ha, D., Dai, A. and Le, Q., 2016. Proceedings of the International Conference on Learning Representations.

37. **Separating style and content with bilinear models** [link]
Tenenbaum, J.B. and Freeman, W.T., 2000. Neural Computation, Vol 12(6), pp. 1247--1283. MIT Press.

38. **Visualizing data using t-SNE** [PDF]
Maaten, L.v.d. and Hinton, G., 2008. Journal of machine learning research, Vol 9(Nov), pp. 2579--2605.

39. **A dataset and architecture for visual reasoning with a working memory** [PDF]
Yang, G.R., Ganichev, I., Wang, X., Shlens, J. and Sussillo, D., 2018. arXiv.

40. **A parallel computation that assigns canonical object-based frames of reference** [link]
Hinton, G.F., 1981. Proceedings of the International Joint Conference on Artificial Intelligence.

41. **The correlation theory of brain function** [link]
von der Malsburg, C., 1994. Models of Neural Networks: Temporal Aspects of Coding and Information Processing in Biological Systems, pp. 95--119.

42. **Generating text with recurrent neural networks** [link]
Sutskever, I., Martens, J. and Hinton, G., 2011. Proceedings of the International Conference on Machine Learning.

43. **Robust boltzmann machines for recognition and denoising** [PDF]
Tang, Y., Salakhutdinov, R. and Hinton, G., 2012. IEEE Conference on Computer Vision and Pattern Recognition.

44. **Factored conditional restricted Boltzmann machines for modeling motion style** [link]
Taylor, G.W. and Hinton, G.E., 2009. Proceedings of the International Conference on Machine Learning.

45. **Combining discriminative features to infer complex trajectories** [PDF]
Ross, D.A., Osindero, S. and Zemel, R.S., 2006. Proceedings of the International Conference on Machine Learning.

46. **Learning where to attend with deep architectures for image tracking** [link]
Denil, M., Bazzani, L., Larochelle, H. and de Freitas, N., 2012. Neural Comput., Vol 24(8), pp. 2151--2184.

47. **Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis** [link]
Le, Q.V., Zou, W.Y., Yeung, S.Y. and Ng, A.Y., 2011. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

48. **Convolutional learning of spatio-temporal features** [link]
Taylor, G.W., Fergus, R., LeCun, Y. and Bregler, C., 2010. ECCV.

49. **Learning to relate images** [PDF]
Memisevic, R., 2013. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 35(8), pp. 1829-1846.

50. **Incorporating side information by adaptive convolution** [PDF]
Kang, D., Dhar, D. and Chan, A., 2017. Advances in Neural Information Processing Systems.

51. **Learning multiple visual domains with residual adapters** [PDF]
Rebuffi, S., Bilen, H. and Vedaldi, A., 2017. Advances in Neural Information Processing Systems.

52. **Predicting deep zero-shot convolutional neural networks using textual descriptions** [PDF]
Ba, J., Swersky, K., Fidler, S. and Salakhutdinov, R., 2015. Proceedings of the IEEE International Conference on Computer Vision.

53. **Zero-shot task generalization with multi-task deep reinforcement learning** [PDF]
Oh, J., Singh, S., Lee, H. and Kohli, P., 2017. Proceedings of the International Conference on Machine Learning.

54. **Separating style and content** [PDF]
Tenenbaum, J.B. and Freeman, W.T., 1997. Advances in neural information processing systems.

55. **Facial expression space learning** [link]
Chuang, E.S., Deshpande, F. and Bregler, C., 2002. Proceedings of the Pacific Conference on Computer Graphics and Applications.

56. **Personalized recommendation on dynamic content using predictive bilinear models** [PDF]
Chu, W. and Park, S., 2009. Proceedings of the International Conference on World Wide Web.

57. **Like like alike: joint friendship and interest propagation in social networks** [PDF]
Yang, S., Long, B., Smola, A., Sadagopan, N., Zheng, Z. and Zha, H., 2011. Proceedings of the International Conference on World Wide Web.

58. **Matrix factorization techniques for recommender systems** [PDF]
Koren, Y., Bell, R. and Volinsky, C., 2009. Computer, Vol 42(8). IEEE.

59. **Bilinear CNN models for fine-grained visual recognition** [PDF]
Lin, T., RoyChowdhury, A. and Maji, S., 2015. Proceedings of the IEEE International Conference on Computer Vision.

60. **Convolutional two-stream network fusion for video action recognition** [PDF]
Feichtenhofer, C., Pinz, A. and Zisserman, A., 2016. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

61. **Multimodal compact bilinear pooling for visual question answering and visual grounding** [PDF]
Fukui, A., Park, D.H., Yang, D., Rohrbach, A., Darrell, T. and Rohrbach, M., 2016. Proceedings of the Conference on Empirical Methods in Natural Language Processing.

Updates and Corrections

If you see mistakes or want to suggest changes, please create an issue on GitHub.

Reuse

Diagrams and text are licensed under Creative Commons Attribution CC-BY 4.0 with the source available on GitHub, unless noted otherwise. The figures that have been reused from other sources don’t fall under this license and can be recognized by a note in their caption: “Figure from …”.

Citation

For attribution in academic contexts, please cite this work as

```
Dumoulin, et al., "Feature-wise transformations", Distill, 2018.
```

BibTeX citation

```
@article{dumoulin2018feature-wise,
  author = {Dumoulin, Vincent and Perez, Ethan and Schucher, Nathan and Strub, Florian and Vries, Harm
de and Courville, Aaron and Bengio, Yoshua},
  title = {Feature-wise transformations},
  journal = {Distill},
  year = {2018},
  note = {https://distill.pub/2018/feature-wise-transformations},
  doi = {10.23915/distill.00011}
}
```

Distill is dedicated to clear explanations of machine learning

About Submit Prize Archive RSS GitHub Twitter ISSN 2476-0757