Language Learning as Meta-Learning

Jacob Andreas **Dan Klein** **Sergey Levine**
Computer Science Division
University of California, Berkeley
{jda,klein,svlevine}@eecs.berkeley.edu

Abstract

Can background knowledge from language improve the generality and efficiency of learned models? We present a model that uses the space of natural language strings as a *parameter* space to capture natural task structure. Crucially, our approach does not require language data to learn new concepts: language is used only in pretraining to impose structure on subsequent learning. Results on image classification, text editing, and reinforcement learning show that, in all settings, models with a linguistic parameterization outperform those without.

The structure of natural language reflects the structure of the world. For example, the fact that it is easy for us to communicate the concept *left of the circle* but comparatively difficult to communicate *mean saturation of the first five pixels in the third column* reveals something about the kinds of abstractions we find useful for interpreting and navigating our environment (Gopnik and Meltzoff, 1987). One of the primary goals of meta-learning is to discover reusable computational abstractions to enable efficient learning on individual tasks. This paper investigates whether background knowledge from language can provide a useful scaffold for acquiring such abstractions. We specifically propose to use language as a latent parameter space for few-shot learning problems of all kinds

In our work, the product of meta-learning is a language interpretation model that maps from natural language descriptions to concepts (e.g. classifiers or transducers). New concepts are learned by searching directly in the space of natural language strings to minimize the loss incurred by the interpretation model (Figure 1). Especially on tasks that require the learner to successfully model some high-level compositional structure shared by the training examples, natural language hypotheses serve a threefold purpose: they make it easier to discover these compositional concepts, harder to overfit to few examples, and easier to understand inferred patterns.

We find that the structure imposed by a natural-language parameterization is helpful for efficient learning and exploration. The approach outperforms more standard multitask- and meta-learning approaches that map directly from training examples to outputs by way of a real-valued parameterization, as well as approaches that make use of natural language annotations as an additional supervisory signal rather than an explicit latent parameter. The natural language concept descriptions inferred by our approach often agree with human annotations when they are correct, and provide an interpretable debugging signal when incorrect. In short, by equipping models with the ability to "think out loud" when learning, they become both more comprehensible and more accurate.

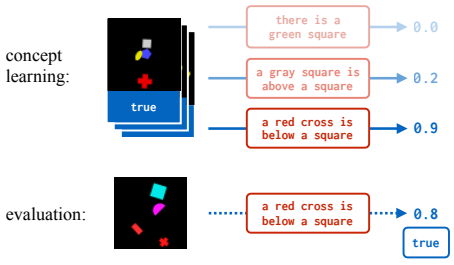

Figure 1: Example of our approach on binary image classification. In meta-training (not shown), we learn a language interpretation model that outputs the probability that an image matches a description. To learn a new visual concept, we optimize over descriptions to maximize the interpretation model's score. The chosen description can be used to classify new images.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

1 Background

Suppose we wish to solve an image classification problem that involves mapping from images x to binary labels y. One straightforward approach is to solve a learning problem of the form $\arg\min_\eta \sum_{x,\,y} L(f(x;\,\eta),\,y)$, where L is a loss function and f is a class of parametric models (e.g. convolutional networks) indexed by η (e.g. weight matrices) that map from images to labels. Given a new image x', $f(x';\,\eta)$ can then be used to predict its label. In the present work, we are particularly interested in *few-shot* learning problems where the number of $(x,\,y)$ pairs is small—on the order of five or ten examples. Under these conditions, directly solving the optimization problem above is a risky proposition—any model class powerful enough to capture the true relation between inputs and outputs is also likely to overfit. For few-shot learning to be successful, extra structure must be supplied to the learning algorithm. Multitask (Caruana, 1998) and meta-learning approaches (e.g. Schmidhuber, 1987; Santoro et al., 2016; Vinyals et al., 2016) attempt to import this structure from collections of other related learning problems, with a learning process that takes place in three phases:

1. a **pretraining** (or "meta-training") phase that makes use of various different datasets i with examples $\{(x_1^{(\ell i)}, y_1^{(\ell i)}), \ldots, (x_n^{(\ell i)}, y_n^{(\ell i)})\}$
2. a **concept-learning** phase in which the pretrained model is adapted to fit data $\{(x_1^{(c)}, y_1^{(c)}), \ldots, (x_n^{(c)}, y_n^{(c)})\}$ for a specific new task
3. an **evaluation** phase in which the learned concept is applied to a new $x^{(e)}$ to predict $y^{(e)}$

Learning operates over two collections of real-valued weights: shared parameters η and task-specific parameters θ. In this view, multitask approaches learn η some subset of θ at pretraining time, then optimize new $\theta^{(c)}$ at concept-learning time; meta-learning approaches learn to predict $\theta^{(c)}$ directly.

2 Learning with Language

In this work, we instead propose to parameterize models with natural language. Natural language has a number of advantages over the raw real-valued parameterization used by existing approaches to few-shot learning: it has a rich set of compositional operators and comes equipped with a natural description length prior, while still exhibiting considerable expressive power. The set of primitive operators available in language provides a great deal of information about the kinds of abstractions that are useful for natural learning problems. Our thesis is thus that language learning is a powerful, general-purpose kind of meta-learning, even for tasks that do not directly involve language.

We call our approach **learning with latent language** (L^3). Concretely, we take the meta-learning phase above to be a **language-learning** phase. We assume that at meta-learning time we additionally have access to natural-language **descriptions** $w^{(\ell i)}$. We use these w as *parameters*, in place of the task-specific parameters θ—that is, we learn a language **interpretation** model $f(x;\,\eta,\,w)$ that uses weights η to turn a description w into a function from inputs to outputs. In the case of image classification, f might be an image rating model (Socher et al., 2014) that outputs a scalar judgment y of how well an image x matches a caption w.

Because these natural language parameters are observed at meta-learning time, we initially need only learn the real-valued shared parameters η used for their interpretation (e.g. the weights of a neural network that implements the image rating model):

$$\arg\min_{\eta \in \mathbb{R}^a} \quad \sum_{i,\,j} L\big(f(x_j^{(\ell i)};\,\eta,\,w^{(\ell i)}),\,y_j^{(\ell i)}\big)\,. \tag{1}$$

At concept-learning time we solve only the part of the optimization problem over strings:

$$\arg\min_{w' \in \Sigma^*} \quad \sum_{j} L\big(f(x_j^{(c)};\,\eta,\,w^{(c)}),\,y_j^{(c)}\big)\,. \tag{2}$$

Because this last step involves optimization over a discrete space of strings, we cannot use gradient descent; we instead rely on learning to help us develop an effective optimization procedure for natural language parameters (Devlin et al., 2017). In particular, we simply use the meta-learning data to fit a reverse **proposal** model, estimating $\arg\max_\lambda \log q(w_i|\{x_j^{(\ell i)}, y_j^{(\ell i)}\};\lambda)$, where q provides a

(suitably normalized) approximation to the distribution of descriptions given training examples. In the running example, this proposal distribution is essentially an image captioning model (Donahue et al., 2015). By sampling from q, we expect to obtain candidate descriptions that are likely to obtain small loss. But our ultimate inference criterion is still the true model f: at evaluation time we perform the minimization in Equation 2 by drawing a fixed number of samples, selecting the hypothesis $w^{(c)}$ that obtains the lowest loss, and using $f(x^{(e)};\, \eta,\, w^{(c)})$ to make predictions.

What we have described so far is a generic procedure for equipping collections of related learning problems with a natural language hyptothesis space. Below we describe how this procedure can be turned into a concrete algorithm for supervised classification and sequence prediction.

3 Few-shot Classification

We begin by investigating whether language can be used to support high-dimensional few-shot classification. Our focus is on visual reasoning tasks like the one shown in Figure 2. In these problems, the learner is presented with four images, all positive examples of some visual concept like *a blue shape near a yellow triangle*, and must decide whether a fifth, held-out image matches the same concept.

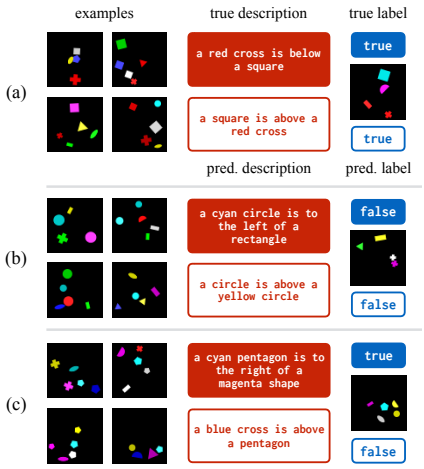

Figure 2: Example predictions for image classification. The model achieves high accuracy even though predicted descriptions rarely match the ground truth. High-level structure like the presence of certain shapes or spatial relations is consistently recovered.

To apply the recipe in Section 1, we need to specify an implementation of the interpretation model f and the proposal model q. We begin by computing representations of input images x. We start with a pre-trained 16-layer VGGNet (Simonyan and Zisserman, 2014). Because spatial information is important for these tasks, we extract a feature representation from the final convolutional layer of the network. This initial featurization is passed through two fully-connected layers to form a final image representation $\texttt{rep}(x)$. Then we define interpretation and proposal models: $f(x; w) = \sigma\big(\texttt{rnn-enc}(w)^\top \texttt{rep}(x)\big)$; $q(w \mid \{x_j\}) = \texttt{rnn-dec}\big(w \mid \frac{1}{n}\sum_j \texttt{rep}(x_j)\big)$. The interpretation model f outputs the probability that x is assigned a positive class label, and is trained using negative log likelihood. Because only positive examples are provided in each language learning set, the proposal model q can be defined in terms of inputs alone.

Our evaluation aims to answer two questions. First, does the addition of language to the learning process provide any benefit over ordinary multitask or meta-learning? Second, is it specifically better to use language as a hypothesis space for concept learning rather than just an additional signal for pretraining? We use several baselines to answer these questions: (1) *Multitask*: a multitask baseline in which the definition of f above is replaced by $\sigma(\theta_i^\top \texttt{rep}(x))$ for task-specific parameters θ_i; (2) *Meta*: a meta-learning baseline in which f is defined by $\sigma([\frac{1}{n}\sum_j \texttt{rep}(x_j)]^\top \texttt{rep}(x))$; (3) *Meta+Joint*: as in *Meta*, but the pretraining objective includes an additional term for predicting q (discarded at concept-learning time).

Model	Val	Test
Random	50	50
Multitask	57	59
Meta	62	64
Meta+Joint	66	64
L^3 (ours)	**71**	**70**
L^3 (oracle)	79	78

Table 1: Results for classification.

We report results on a dataset derived from the ShapeWorld corpus of Kuhnle and Copestake (2017). Results are shown in Table 1. It can be seen that L^3 provides consistent improvements over the baselines, and that these improvements are present both when identifying new instances of previously-learned concepts and when discovering new ones. Some example model predictions are shown in Figure 2 (more in Appendix B). The model often succeeds in making correct predictions, even though its inferred descriptions rarely match the ground truth. Sometimes this is because of inherent ambiguity in the description language (Figure 2a), and sometimes because the model is able to rule out candidates on the basis of partial captions alone (Figure 2b, where it is sufficient to recognize that the target concept involves a *circle*).

4 Programming by Demonstration

Next we explore whether the same technique can be applied to tasks that involve more than binary similarity judgments. We focus on structured prediction: specifically a family of string processing tasks. In these tasks, the model is presented with five strings being transformed according to some rule; it must then apply an appropriate transformation to a sixth (Figure 3). Learning proceeds more or less as in the previous section, with $\texttt{rep}(x, y) = \texttt{rnn-enc}([x, y])$; $f(y \mid x; w) = \texttt{rnn-dec}\big(y \mid [\texttt{rnn-enc}(x), \texttt{rnn-enc}(w)]\big)$; $q(w \mid \{(x_j, y_j)\}) = \texttt{rnn-dec}\big(w \mid \frac{1}{n}\sum_j \texttt{rep}(x_j, y_j)\big)$.

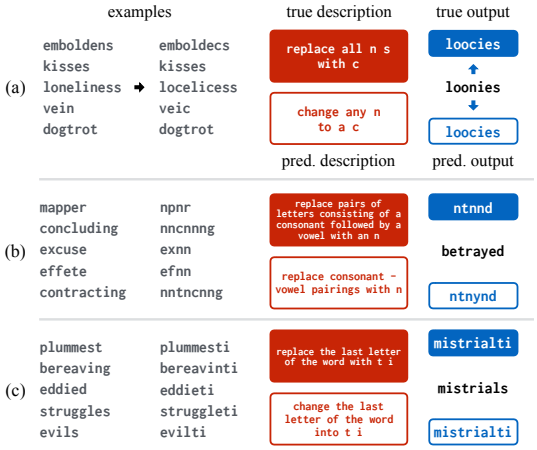

Figure 3: Example predictions for string editing.

For these experiments we have created a new dataset of string editing tasks by (1) sampling random regular transducers, (2) applying these transducers to collections of dictionary words, and (3) showing the collected examples to Mechanical Turk users and asking them to provide a natural language explanation with their best guess about the underlying rule.

Results are shown in Table 2. In these experiments, all models that use descriptions have been trained on the natural language supplied by human annotators. While we did find that the Meta+Joint model converges considerably faster than all the others, its final performance is somewhat lower than the baseline Meta model. As before, L^3 outperforms alternative approaches for learning directly from examples with or without descriptions. More examples are given in Appendix B.

Model	Val	Test
Identity	18	18
Multitask	54	50
Meta	66	62
Meta+Joint	63	59
L^3	**80**	**76**

Table 2: Results for string editing.

Because all of the transduction rules in this dataset were generated from known formal descriptors, these tasks provide an opportunity to perform additional analysis comparing natural language to more structured forms of annotation (since we have access to ground-truth regular expressions) and more conventional synthesis-based methods (since we have access to a ground-truth regular expression execution engine). A few interesting facts stand out. With no ground-truth annotations provided, meta-learning with natural language data gives an accuracy of 80%, while regular expression data gives an accuracy of 76%—natural language is actually better than precise structured descriptions! This might be because the extra diversity helps the model figure out the relevant axes of variation and avoid overfitting to individual strings. Allowing the model to do its own inference is also better than providing ground-truth natural language descriptions (80% vs 75%), suggesting that it is actually better at generalizing from the relevant concepts than human annotators (who occasionally write things like *I have no idea* for the inferred rule). Coupling our inference procedure with an oracle RE evaluator, we essentially recover the synthesis-based approach of Devlin et al. (2017). Our findings are consistent with theirs: when a complete and accurate execution engine is available, there is no reason not to use it. But we can get almost 90% of the way there with an execution model learned from scratch. Some examples of model behavior are shown in Figure 3.

5 Conclusion

We have presented an approach for optimizing models in a space parameterized by natural language. Using standard neural encoder–decoder components to build models for representation and search in this space, we demonstrated that our approach outperforms strong baselines on classification, structured prediction and reinforcement learning tasks.

We believe the key lesson of this work is that *language encourages compositional generalization*: standard deep learning architectures are good at recognizing new instances of previously-encountered concepts, but not always at generalizing to new ones. By forcing decisions to pass through a linguistic bottleneck in which the underlying compositional structure of concepts is explicitly expressed, stronger generalization becomes possible.

References

Caruana, R. (1998). Multitask learning. In *Learning to learn*. Springer.

Devlin, J., Uesato, J., Bhupatiraju, S., Singh, R., Mohamed, A.-r., and Kohli, P. (2017). RobustFill: Neural program learning under noisy I/O. In *Proceedings of the International Conference on Machine Learning*.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*.

Gopnik, A. and Meltzoff, A. (1987). The development of categorization in the second year and its relation to other cognitive and linguistic developments. *Child Development*.

Kuhnle, A. and Copestake, A. (2017). Shapeworld-a new test methodology for multimodal language understanding. *arXiv preprint arXiv:1704.04517*.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. In *Proceedings of the International Conference on Machine Learning*.

Schmidhuber, J. (1987). Evolutionary principles in self-referential learning. *On learning how to learn. Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arxiv:1409.1556*.

Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., and Ng, A. Y. (2014). Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*.