Using prototypes to improve convolutional networks interpretability

Thalita Drumond, Thierry Viéville, Frédéric Alexandre

Using prototypes to improve convolutional networks interpretability

Thalita F. Drumond **Thierry Viéville** **Frédéric Alexandre**
thalita.firmo-drumond@inria.fr
Mnemosyne Team, INRIA Bordeaux Sud-Ouest, Labri UMR 5800, IMN UMR 5293, France

Abstract

We propose a method that allows the interpretation of the data representation obtained by CNN, through introducing prototypes in the feature space, that are later classified into a certain category. This way we can see how the feature space is structured in link with the categories and the related task.

1 Introduction

Image recognition performance has been highly boosted thanks to convolutional neural nets (CNN) (e.g. [12, 19, 16, 6]). Such models are able to infer relevant features and discriminate categories, but learning relies on the availability of sufficiently large datasets, while the result is hardly interpretable. We claim that better interpreting CNN's large dimensional representations (i.e., data embedding) helps better understanding the underlying decision-making process, and propose to adapt the notion of "prototypes" in order to describe category mapping in the embedding space.

In the brain, the infero-temporal cortex has a kind of prototypical representation via neurons tuned to respond to different categories. See, e.g. [20], including the link between machine learning algorithms and biological models. This latter stage of the visual cortex hierarchy motivates our use of a new prototypical approach in convolutional networks.

Inspired by this knowledge we propose a new model, that not only describes samples in the feature space through prototypes, but also deals with classes scattered in this feature space, using more than one prototype. Our flexible mapping allows the identification of ambiguous classification regions in the feature space, with prototypes in link with several classes. This may correspond to new intermediate categories or regions sensitive to adversarial perturbations. Within our framework, singleton and unbalanced classes are taken in to account. Altogether, this helps investigating how the feature space is structured with respect to classes, for a given task. Finally, we consider here a few-shot learning paradigm (e.g., a few samples by category, thanks to transfer learning). Prototype-based models have indeed already be proposed [17, 13], with efficient results, but they cannot treat scenarios such as discussed in [18], where features are still twisted, with a class represented by multiple disconnected components in the feature space.

2 Related works

The term prototypes is used in the literature with different connotations: a priory information, clustering representation, quantification of space, as discussed now.
The prototypical priors layer proposed in [10], where prototype images of road signs, chosen a priory, are encoded using a HoG (histogram of oriented gradients) descriptor, and added as fixed units of the penultimate layer. This way the network is ultimately trained to match the HoG representation of the prototypes. However this work assumes the existence of a standard representative image per class in the input space to be encoded as a prototype unit.
In prototypical networks [17], prototypes correspond to class centroids in the feature space spanned by the embedding CNN. Then the nearest-prototype is used to classify a given sample. This approach

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

is simple and proved effective on the considered datasets. However, it does not respond to the scenario of metric learning proposed by [18]. Deriving from this work there are also Gaussian prototypical networks [5], which predict a covariance radius for each prototype, yielding some insight on the discriminating force of each one of them.
The prototypical sampling on the data space [7], as performed by self-organizing maps, or the dictionary sparse representation methods [14], allows the representation of what is known about a data distribution, using methods quoted in prototypical networks and known as optimizing statistical criteria [1].

Then, regarding the few-shot learning challenge, having access to large databases is unfeasible in some domains and specific applications. This motivates the development of models that can use the representation power of convolutional nets, but still learn on few to single examples of each category to be recognized, this mainly by transfer learning [22].
Few-shot learning has recently been established as a generalization of the one-shot learning task, and refers to the paradigm where the model has to make predictions over N new classes given only a small number K of examples per class, often as additional data on other classes are used to pre-train the model prior to tackling the smaller K-shot N-way dataset. Many strategies have been proposed to deal with this new task, including [11, 21, 17, 15].
Other numerous strategies are derived from metric learning [2] and deep embedding learning. These methods have in common the use of a neural network model to learn and the embedding of the input data generating features on a metric space. The objective function usually minimizes the distance between pairs of the same class, like with siamese networks [11]), while one may also consider triples to simultaneously push apart a third example of a different class.
Distance-based image classification is highly related to our approach. In [13] two distance-based classifiers, the k-nearest neighbor (k-NN) and nearest class mean (NCM) classifiers are considered, introducing a new metric learning approach for the NCM, allowing to test a multiple prototype per class alternative, initialized by k-means.
Finally deep metric learning via Facility Location [18], brings up the fact that deep networks learn local metrics so that we can have groups of examples of a same class pushed far apart from each other when there are examples of other classes in between, e.g. using a loss function based on a clustering quality metric, here normalized mutual information.

3 Model proposed

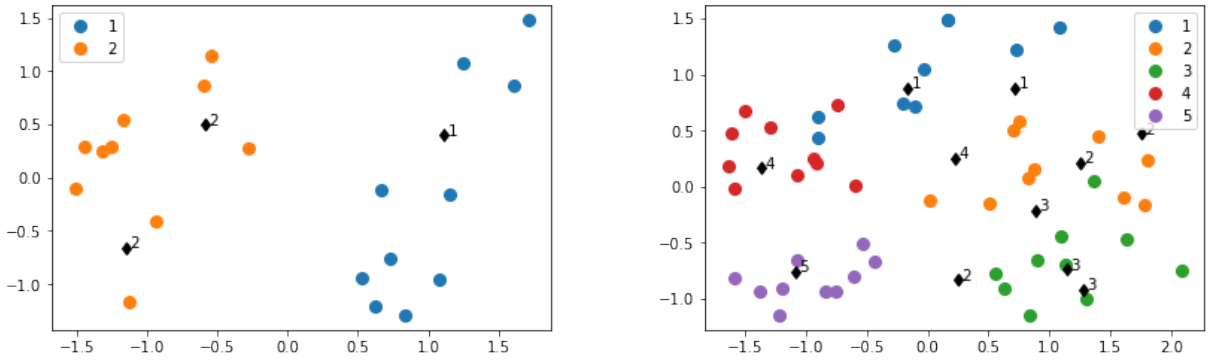
Model architecture. In order to improve the previous proposals, after the embedding provided by a CNN, we introduce a prototype matching layer as proposed in [17], defined by a set of prototypes sampling the input distribution in the feature space. This layer is a soft-max applied to sample to prototype proximity, allowing competition between prototypes, grouping examples with respect to the induced metric learned by the CNN layers, and outputting a matching probability for each prototype. Improving this idea, this output feeds an additional final soft-max classifier layer, that correlates the prototype's matching to class labels, introducing a sparse estimation in order to encourage prototype to class label assignment. In order to perform such features discrimination, we neither hard-wire sample to prototype matching nor prototype to class label assignment, but let the estimation provide the best description of the data, given the classification task, in an interpretable way discussed below.

Model specification. We consider a set of labeled images $\{\cdots(\mathbf{I}_i, l_i)\cdots\}, i \in \{0, M\{$, labels being finite $l_i \in \{0, K\{$. This input space $\mathcal{I}$ is embedded in a feature space $\mathcal{X}$ of huge dimension D through a function $f : \mathcal{I} \to \mathcal{X}$, here defined by the CNN. We introduce the notion of prototype in order to discriminate between twisted features at the output of some CNN. To this end, we propose to consider the estimation of a fixed-size set of prototypes in the feature space $\mathbf{p}_j$, $j \in \{0, J\}, J \geq K$, in two steps:.
Here, $q_j(\mathbf{x}) = p(\mathbf{p}_j \in Q_j|\mathbf{x})$ is the probability for the feature vector $\mathbf{x}$ of a sample, to be associated to the j-th prototype. This association is written as $\mathbf{x} \in Q_j$, where Q_j is the set of samples associated to the given prototype. Interestingly enough, our model does not implicitly assume that the Q_j forms a partition of the space, ad easily proved by comparing with hard partition Bayesian rule. These regions may overlap or uncover the whole space. Furthermore a sample may be sparsely represented, thanks to the soft-max criterion, by several prototypes as in a dictionary representation method (see e.g. [14]). We approximate $q_j(\mathbf{x})$ as a function of the proximity $-\|\mathbf{x}-\mathbf{p}_j\|$ to the prototypes, writing: $q_j(\mathbf{x}) = \nu_j\left(-\|\mathbf{x}-\mathbf{p}_j\|^2/2\right)$, where $\nu_j(\cdot)$ stands for the soft-max distribution, the related $N \times J$

weight matrix of this layer corresponding to prototypes coordinates in the feature space $\mathcal{X}$.
Then, $c_l(\mathbf{x}) = p(l|\mathbf{x})$ is the probability for a sample to be associated to the label of index l, estimated by another soft-max layer: $c_l(\mathbf{x}) = \nu_l \left(\mathbf{w}_l^T \mathbf{q}(\mathbf{x})\right)$, the related $J \times L$ weight matrix corresponding to the prototype-to-class association. Considering an approximate cross-entropy criterion to adjust the parameters given samples $\mathbf{x}_i$ and their label l_i yields the following minimization: $\mathcal{C} = -\int_{\mathcal{X}} p(\mathbf{x}) \log\left(c(\mathbf{x})\right) + \alpha |\mathbf{w}_l|_1 \simeq -\frac{1}{N}\sum_i \log\left(c_{l_i}(\mathbf{x}_i)\right) + \alpha |\mathbf{w}_l|_1$. A straightforward derivation of the related normal equations leads to an estimation-minimization method:
The $\partial_{\mathbf{p}_j}\mathcal{C} = 0$ equation yields a k-mean estimation of the prototypes given the samples, weighted by the prototype proximity. This deep relation between divergence estimation and k-mean algorithms is known [1] and will allow us to derive an efficient implementation, and to minimize the role of meta-parameters for this step.
The $\partial_{\mathbf{w}_l}\mathcal{C} = \sum_i \left[\delta_{l=l_i} - c_l(\mathbf{x}_i)\right] \mathbf{q}(\mathbf{x}_i) + \mathcal{R}$, where $\mathcal{R}$ stands for regularization terms, yields a Hebbian-like interpretable 1st order rule for adjusting the prototype-to-class association, as a function of each training sample a-priory class label. The meta-parameters of this part of the estimation may be automatically tuned, e.g., the α controlling the sparseness can be integrated in the optimization process, either using Jeffreys Prior [4] or simply exploring different values in order to obtain the sparsest value without generating additional misclassification.

4 Simulations and results

As a proof-of-concept, we applied our model to pre-trained CNN features (namely from Inception V3 network [19] with ImageNet weights), extracted from the Omniglot dataset. It consists of images of 1622 characters from different alphabets, each drawn by 20 different people. The CNN features have 2048 dimensions. Experiments are run over a subset of N classes, chosen at random from the 1622 available classes. Results are averaged over 10 different subsets of classes. For each subset, the 20 samples per class get split in two folds – training and test. The training fold is again divided in 2 for cross-validation over hyper-parameters (regularization weight and number of prototypes). Thus we perform a 5-shot (i.e. 5 training samples per class) N-way learning task, for N=2, 5, and 20 classes. As a comparison, we perform the same task with a one prototype per class approach (like [17] and also with a softmax classifier with L2 regularization. Convergence is reached for all methods (within 10^{-4} tolerance when using L2 and 10^{-1} when using L1).



(a) 2 class example

(b) 5 class example

Figure 1: Classified samples projected on 2D (via PCA). Colors indicate true class. Prototypes are indicated by a black diamond annotated with the winning class id.

Table 1: Accuracy on test set for 5-shot task. Mean(std. dev.) over 10 class subsets are shown.

	Our model		Single-prototype	Softmax
Regularization	L1	L2	–	L2
2-way	0.80(0.11)	**0.91(0.08)**	**0.93(0.04)**	0.79(0.14)
5-way	0.81(0.07)	0.90(0.04)	0.91(0.04)	**0.96(0.02)**
20-way	0.52(0.10)	0.74(0.04)	0.77(0.03)	**0.89(0.02)**

As observed in Fig. 1, the algorithm correctly clusters the data and provide a summarized description of the samples as a set of prototypes. Accuracy results, reported in Table 1, are comparable to the single-prototype method. According to a standard t-test, our results are not distinguishable from the single-prototype approach (with $P_{H_1} \simeq 0.5$), for all N=2, 5 and 20. Interesting enough is the fact that a L2 regularization yields better results than considering sparseness using L1: The assumption that a prototype may be related to several class seems numerically relevant, as it can be observed in Fig. 1 (as far as the PCA projection is significant) where some twisted regions with more than one category are still represented by prototypes. Both approaches are still to be improved when considering a higher number of classes.

5 Discussion

This method produces insight into existing CNN models, allowing us to interpret the representations learned and decisions made by the model. The key point is that prototypes are neighborhood representatives on the feature space. Such prototype representation is in one to one correspondence with the class segmentation of the data space, as discussed in e.g., [3]. The network "weights" have a clear explicit interpretation in terms of prototype feature coordinates and prototype class label association. The latter provides the knowledge of which prototypes compose each class. These estimated parameters allows us to identify scattered classes defined as a disjunction of remote prototypes, ambiguous classification, when prototypes are, e.g., equally related to several categories, and occurrence of some singleton. From a safety point of view, it also conveys information on susceptibility to adversarial perturbations, which are minimal changes to the input yielding misclassification [9]. This can be made explicit as soon as such perturbations are included in the validation dataset.
The former penultimate layer allows to instance virtual examples explicitly by reconstructing the maximizing input for each prototype (e.g., [23]), and describing the data space by exploring the sample topography in the neighborhood the prototypes (e.g., using a PCA to provide a graphical representation as schematized in Fig.1), or computing any application relevant statistical parameters (e.g., Gaussian 2nd order representation of each prototype). For these reasons, and because we discourage but do not forbid samples of different categories to match a given prototype, our prototypes are more than category representatives, acting as neighborhood descriptors. Finally, the possibility of reconstructing representative examples from prototypes is easier to explain to a lay audience, being a means to provide an explanation for algorithmic decisions that may affect citizen life.

6 Perspectives

More generally, interpretability of a model can arise during (i) learning and (ii) inference, not only by analyzing network parameters. The latter aspect includes not only classifying a sample but also gaining knowledge on its neighborhood in the feature space, or the level of ambiguity of its classification. These are perspectives of this work.
Beyond discussing interpretability of such approach, let us point out how the fact we can easily interpret such processing allows us to propose several improvements.
We can not only observe the local metric induced by prototypes but, generalizing [18], propose to improve the related clustering quality metric, readjusting the CNN weights in order to untwist the obtained representation (pull apart/closer prototypes associated to different/identical labels). In other words, we propose not only to consider the deep learning architecture as a feed-forward pipeline but also to introduce well-defined feedback.
Our approach is not limited to the CNN upper layers, the feature space may also be enriched by lower level features, depending on the application, which can be achieved through forward shortcuts (e.g., as in [6, 8]).
The appropriate number of prototypes is, at the present stage, a meta-parameter, which is usually not known in advance. The method might be improved so as to include or remove prototypes incrementally during training, according to the clusters encountered, thus adjusting automatically the prototype count.
Future experiments also include integrating the current prototypical layer with a CNN, training the model end-to-end, following the experimental paradigm of [21, 17].

Acknowledgments

Work partially developed under project FUI Sumatra (PIA and Nouvelle Aquitaine Region fund).

References

[1] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[2] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. Technical report, jun 2013.

[3] Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1):1–49, 2002.

[4] Mario A. T. Figueiredo. Adaptive sparseness using Jeffreys' prior. *Advances in neural information processing systems*, 1:697–704, 2002.

[5] Stanislav Fort. Gaussian prototypical networks for few-shot learning on omniglot. *arXiv preprint*, aug 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[7] Thomas Hecht and Alexander Gepperth. Computational advantages of deep prototype-based learning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9887 LNCS:121–127, 2016.

[8] Gao Huang, Zhuang Liu, Kilian Q. Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[9] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *International Conference on Computer Aided Verification*, pages 3–29. Springer, 2017.

[10] Saumya Jetley and Philip Torr. Prototypical priors : From improving classification to zero-shot learning. In *BMVC*, pages 1–12, 2015.

[11] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML - Deep Learning Workshop*, volume 37, 2015.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira Weinberger, C. J. C. Burges, L. Bottou, and K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[13] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2624–2637, nov 2013.

[14] Ron Rubinstein, Alfred M. Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[15] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks. *arXiv preprint*, may 2016.

[16] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICRL)*, pages 1–14, sep 2015.

[17] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017.

[18] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5382–5390, dec 2017.

[19] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, sep 2015.

[20] Thierry Viéville and Sylvie Crahay. Using an hebbian learning rule for multi-class svm classifiers. *Journal of Computational Neuroscience*, 17(3):271–287, 2004.

[21] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *arXiv preprint*, jun 2016.

[22] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. *A survey of transfer learning*, volume 3. Springer International Publishing, 2016.

[23] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, 8689:818–833, 2014.