Stacked Convolutional Auto-encoders for Hierarchical Feature Extraction

O. Chen, D. Simig, G. Weisz

Department of Engineering,
University of Cambridge

Introduction

This work investigates the **convolutional auto-encoder** (CAE) as an unsupervised learning model for extracting hierarchical features from natural images. A CAE is similar to a traditional auto-encoder except it uses convolutional layers for the hidden layers in the network. We show in our experiment replications on MNIST that this model is capable of learning robust feature-representations for image data. Furthermore, it is possible to use the weights learned by a (stacked) CAE to initialize the weights in a **convolutional neural network** for classification tasks.

Preliminaries

Classically, a neural network is a highly non-linear function that maps input vectors to output classes/categories. They typically consist of an input layer, one or more hidden layers, and an output layer. The recent success of neural networks is largely due to the implementation of deep topologies where many hidden layers are used to capture high-level feature representations for the inputs. **Convolutional neural networks** (CNN) extends the classic neural network by using multiple stacks of convolutional (and pooling) layers for the hidden-layers. These networks have been successfully applied for a wide range of computer vision tasks with state-of-the-art results [1, 2, 3, 4]. **Auto-encoders** are popular models for performing unsupervised feature extraction for highly non-linear data. The simplest implementation of an auto-encoder is a simple feed-forward neural network where the learned latent representations are given by the hidden vector $h = \sigma(W_x x + b_x)$ for an activation function σ, weight matrix W_x, and bias b_x.

Their Approach

A **convolutional auto-encoder** (CAE) differs from the traditional auto-encoder model in that it uses a convolutional (and optionally pooling) layer for the hidden layers. The expression for the k^{th} feature map outputted by a convolutional layer is given by:

$$h_k = \sigma(x * W_k + b_k) \tag{1}$$

where σ is the activation function, h_k is the k^{th} feature map, and $*$ denotes the 2D convolution operator. Similar to CNNs, max-pooling can optionally be applied on the feature maps outputted by a convolutional layer – the activation values would then be the max of multiple k by k patches spanning across a given feature map. For highly non-linear data, a CAE can be stacked (CAES) to obtain a deep structure for modelling the data, similar to [5].

The deep topology of a CAES enables each layer of the network to model increasingly abstract latent representations of the input based on the latent output of the previous layer. As a result, CAES offers a powerful model for learning robust hierarchical latent representations for highly-structured inputs such as natural images. Given the similarity in structure between a CAES and popular CNN classification models based on the architecture of AlexNet [1], it follows that the learned weights in a CAES can also be used for initializing the weights in the latter group of networks. The intuition is that this will ensure the weights in the CNN are initially set to sensible values for training with back-propagation. This is further investigated in the Experiments section.

Experiments and Results

For our experiment replications, we first examined the performance of a CAES in a reconstruction task on handwritten digits from MNIST. For purposes of comparison, we trained a CAE with a single convolutional layer and a CAES with 3 convolutional layers. Training was performed in a greedy layer-by-layer fashion similar to the method described in [6] . We found that CAES models trained without any additional constraints tended to either learn the identity mapping or a single prototypical reconstruction for every single input. To counteract this problem, we performed a simple spike noise-injection procedure where 30% of the input pixels were randomly set to 0 or 1. We also found that extending the CAE model with a fully-connected layer (with dropout) in the decoder led to better convolutional filters learned by the network. Reconstruction MSE further improved when we added max-pooling layers to the model, with noticeable qualitative improvements in the filters learned by the network.

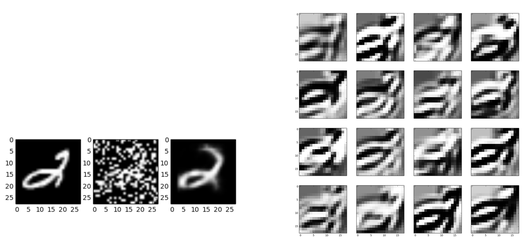

Figure 1: 3-layer CAE model with a noise-injected input. Left: the image at the far-left is the clean image; the image in the middle is the noise-injected input; and the image at the far-right is the reconstructed image. Right: activations in the final convolutional layer before the reconstruction/decoding layer. We used 50% for the spike-noise injection to illustrate robustness to noise.

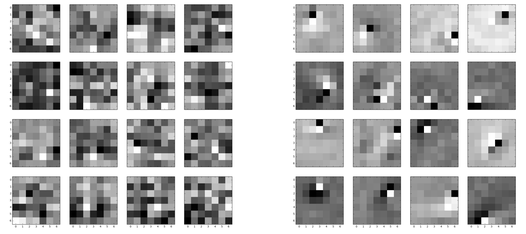

Figure 2: Left: filters learned by CAE with no pooling. Right: filters learned by CAE with pooling. In the case of no-pooling, the filters exhibit seemingly random activation patterns. With the addition of pooling, the filters show a distinct pattern for performing feature detection.

In our second set of experiments, we performed weight-initialization in a CNN with the encoding weights learned from a topologically-equivalent CAES, followed by fine-tuning of the entire network. These experiments were conducted for CAES/CNN models with and without pooling layers. For our baseline, we simply trained a CNN using random weight-initializations. We also experimented with performing fine-tuning only in the classification layer (while keeping all other weights fixed) to evaluate the robustness of the learned features for classification. This was essentially implemented as a linear classifier with the learned representations as inputs. We found that the CNN model still performed reasonably well even without fine-tuning of the weights in the convolutional layer, which demonstrates that the feature representations learned by the CAES were robust and directly applicable for classification. We also observe that max-pooling consistently improved classification performance, regardless of whether fine-tuning was performed for the entire network or just the classification layer. This was unsurprising, since max-pooling performs an implicit form of regularization by restricting the forward propagation of information.

Pooling	Fine-Tuning	MSE	ACC
NO	CL	0.0050	91.1%
NO	FULL	0.0050	92.4%
YES	CL	0.0018	93.0%
YES	FULL	0.0018	98.5%

Table 1: Results from CNN weight-initialization with CAES. First column indicates whether max-pooling was used. Second column indicates whether fine-tuning was performed over the final classification layer (CL) or the entire network (FULL). Third column displays the MSE of the CAES model used for weight initialization. The final column is the percentage of correctly predicted labels for the weight-initialized CNN classification model.

Conclusion

Feature extraction with a CAES is an effective method for obtaining robust feature representations for natural images. CAES outperforms traditional unsupervised learning models in this respect due to its ability to better model spatial localities in images. Indeed, our experiments showed that the features learned by a CAES are directly applicable for classification without the need for further fine-tuning. Moreover, given the similar topology of CAES and CNN models, unsupervised learning in a CAES offers a simple method for weight-initialization in CNNs. We expect this method for weight-initialization to be particularly useful in scenarios where unlabelled data is plentiful but the amount of labelled data is limited. A possible extension of the CAES model would be to investigate how the model can be augmented to handle videos instead of images. In particular, given the temporal relationships which exists between consecutive video frames, it would be interesting to examine whether recurrent neural networks can be incorporated within the overall CAES framework to model these temporal localities. Given the vast availability of unlabelled video data, weight-initialization strategies based on the one discussed here is likely to be crucial for obtaining good performance in video classification tasks.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton.
Imagenet classification with deep convolutional neural networks.
In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[2] Yann LeCun, Fu Jie Huang, and Leon Bottou.
Learning methods for generic object recognition with invariance to pose and lighting.
In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–104. IEEE, 2004.

[3] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al.
What is the best multi-stage architecture for object recognition?
In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.

[4] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng.
Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations.
In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

[5] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol.
Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.
Journal of Machine Learning Research, 11(Dec):3371–3408, 2010.

[6] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber.
Stacked convolutional auto-encoders for hierarchical feature extraction.
In *International Conference on Artificial Neural Networks*, pages 52–59. Springer, 2011.

UNIVERSITY OF CAMBRIDGE
Department of Engineering