Invariant computation model based on computational learning

Akul Gupta: akulgupta2000@gmail.com

May 10, 2020

1 Introduction

Current advances in computer vision technologies, namely using techniques like convolutions nets, have shown great power from digit classification to self driving cars. However, in a goal towards artificial general intelligence, these approaches do not exhibit the "general" intelligence capabilities of the human mind. This paper will outline some of the current shortcomings of current architectures and propose a new architecture that is based on neurophysiological experiments on the human mind.

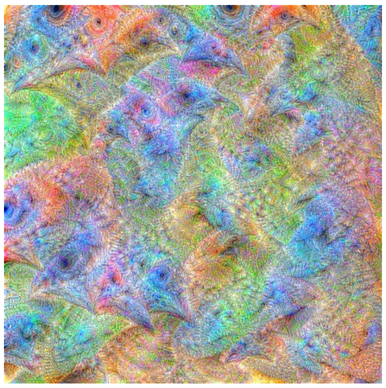

Figure 1: **CNN filter of a "bird head" credit: Fabio M. Graetz**

As an example of lack of generality in current architectures, consider the robustness of visual models to image perturbations. Common architectures like CNN are affected by minimal changes to input images which can dramatically change its classification. The human mind does not face this problem. While this problem can be fixed by adding such perturbation failures to the training set, these approaches go to show the probabilistic nature of these architectures, which are not general in any sense.

Another big problem is the space invariant problem. Fig 1 shows the input image that will result in the maximum activation for a "bird" filter. If you presented the network with a picture of a bird's head in the top right most corner or scaled bigger than the other birds however, this would result in low or no activation this "bird" filter.
If some sort of linear transformation is applied to an object such as scaling or rotation, current architectures are susceptible to failure. Approaches to this problem involve such transformed data to the training or to "convolve the filter on different scales". What these approaches are doing is adding the transformed data to the filter to account for the various transformations.

In the general sense, the distribution of possible scales, orientations, and colors varies dramatically. The human brain does not learn various distributions of such invariant properties for each object as current architectures are doing. Instead the human brain learns the concepts of scale and rotation, and eventually becomes space invariant to such transformations despite only having seen objects in one scale.

2 Abstract

We process information based on relativity. We perceive brightness relative to our surroundings. Our sense of speed is relative to other moving objects. It, therefore, makes intuitive sense to create a model that uses this same principle of relativity. I propose a concept of hierarchical "tensors" which learn compositional features. A tensor is essentially a cortical minicolumn: each tensor has a tuning preference for certain tensors that compose it and they encapsulate all essential information to "activate" that tensor in a single unit (ie size, location, neighborhood). Tensors learn relative relationships which are analogous to hyper-complex cells found in V5, which are selective to certain ratios of stimuli from complex cells. Since tensors are based on relativity, they are space invariant. These tensors also lie in a neighborhood close to similar tensors, mimicking cortical column arrangement in the visual cortex, allowing tensors to generalize. Additionally, these tensors encode properties of size and location: properties that the are present in IT representations ("IT Cortex Contains a General-Purpose Visual Object Representation").

3 Architecture

To model V1, a simple 2 layer CNN with kernel size of 5x5 is used. These will be the unit tensors which higher-dimensional tensors will be composed of. I will refer to tensors that feed into other tensors as input-tensors and the tensor that the input-tensors feed into as the output-tensor.

Tensor representations of regions beyond V1(V2-V6) have the following properties: neighborhood (3D coordinates with z representing the depth of the visual stream), length, center-of-mass, and a relative position. The length of an output-tensor is calculated by summing the length of all the input-tensors. The base case is the unit tensor which has length 1. Center-of-mass is calculated using a simple heuristic which averages the center-of-mass of all input-tensors that compose the output-tensor. The relative position of the input tensors are 2D coordinates. For each input-tensor, they are calculated by finding the distance from the center-of-mass of the output-tensor normalized by the length property of the output-tensor.

In its default state, the output-tensor has a Gaussian parameterization for each input-tensor that composes it, specifically each parameterization for each input-tensor will have four parameters for the neighborhood, length, center-of-mass, and a relative position properties of the input-tensor.

The activation of a tensor is given by the following:

$$Act(T|t_a) = \sum_{\{t_j\}} sim(t_a|t_j)$$

Where t_j are the learnt input-tensors that make up tensor T and $\{t_a\}$ is the set of test input-tensors. t_a is the test input-tensor that corresponds to to the t_j input tensor.

$$= \sum_{\{t_j\}} \mathcal{N}(\mathrm{t}_a|t_{j\mu}\, t_{j\sigma})$$

Where $t_{j\mu}$ and $t_{j\sigma}$ encode the mean and variance of the neighborhood of the input-tensor t_j. This allows generalization to input-tensors that are in a similar neighborhood to t_j.

Additionally these parameterization can learn different weights for each input-tensor to learn different importance's for different input-tensors. For example, if the input-tensor for a stop sign is the tensor representation for the word "stop" and the octagon tensor, the tensor "stop" is more important. The equation then becomes...todo..

3.1 composition

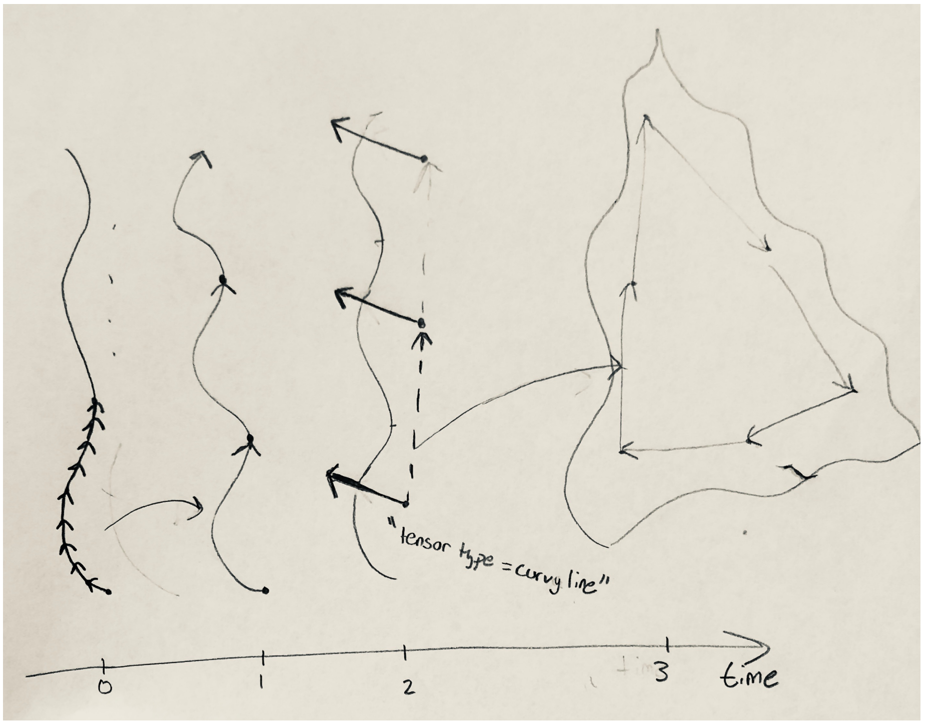

Figure 2: **bottom-up processing of a triangle.**

Consider the following example:
Time t=0 represents early level processing present in the V1 region of the brain. Assume that the granularity of the tensors shown are 5x5. These tensors are unit tensors.

Time t=1 represents the beginning of higher order processing. The curvy line is composed of a collection of tensors from V1. It learns the general position that these tensors are positioned with respect to each other, and the general type of these tensors. In this way, the definition of a curvy line is general. Also note that these tensors are compositionally made.

Time t=2: the vectors coming out of the plane of the figure, ie in the z direction, represent the center of mass property of the curvy line tensors(experiments have shown that we have a visual heuristic for determining center of mass of figures). These center of mass tensors are connected by a "pointer" tensor. Pointer tensors are represented as a straight line tensor. Note the norm of the orthogonal tensors represent the length of the curvy line tensor, with smaller values being associated with smaller curvy lines.

Time t=3: The pointer tensor is now a straight line tensor. These line tensors can be recursively combined to form three larger lines. Thus, three line tensors are created through a recurrence connection.

Time t=4: The definition of a triangle is three line tensors which are arranged in a certain visual-spatial property(ie each line is tangent to two of the other lines). The collection of line tensors in the figure exhibit this visual-spatial property, so it will activate the triangle tensor.

3.2 relativity

Tensors encode properties of relativity. Consider the number "7". It is composed of a horizontal line (the top) and a vertical line (the bottom). It is acceptable for the top line to be equal to or less than the bottom line. However, as the top becomes longer than the bottom, we visual stop seeing it as a "7".

This relationship is easily encoded in the tensor representation of a "7", I will call it Tensor_7.
Tensor_7 is made up two tensors: the horizontal line tensor and vertical line tensor. These tensors both have a respective length which is calculated summing up the the tensors that make it up. The information of length is passed up from the unit tensor (the kernel in the 2 layer CNN in V1), up to the higher level tensors (located deeper in the cortical structure). Tensor_7 will store the information that the top line relative to the bottom line has a ratio of 4/5, with some deviation-which it will learn through the examples it see's. Tensor_7 will therefore be less responsive to a "7" that exhibit large deviations from the learnt relative ratio.

3.3 high/low frequency example

As a higher level example, consider the picture of three triangles. Each triangle tensor has a center of mass tensor and a pointer tensor to the other triangles. After processing these triangles are represented hierarchically, with the overall visual-spatial position of the triangle tensors being encoded at the highest level, followed by the information that those tensors are triangles, followed by the information that those triangles are composed of curly lines.

If you were to look at a mountain, turn away, and sketch the mountain without looking back at it, you would be able to recall the low-frequency details of its general shape and color. You would, however, not be able to recall high-frequency details of its specific texture.

This architecture encapsulates that idea. The low-frequency detail in the example above are the visual-spatial positions of the three triangle. The high-frequency information is the curvy line tensor that the triangles are composed of. In many scenarios where specificity is not needed, the information of curvy lines which often gets ignored. From an efficiency standpoint, it is often pointless to store this high-frequency information.

3.4 Tensor neighborhood

Additionally, tensors have a neighborhood that they lie in that corresponds to their locations in the brain; closely related tensors are grouped closer together (ie a circle and oval tensor will be closely positioned in compared to the triangle tensor). This provides a system for generalization which mimics the structure in the visual cortex where cortical columns next to each other respond to similar features.

For example, consider the two following pictures. While the first figure can easily be classified as a triangle, there is some ambiguity in the second figure. The second figure differs from the first in that it is composed of curvy lines. At the first level in the hierarchy, the figure is composed of curves. These curves make up a curved tensor. These curved tensors however, lie in the pattern of a line, which activates and encodes the repeating curved tensors, as a line tensor. At this point of processing, these line tensors exhibit the property that is most similar to the triangle tensor. Therefore, working up the hierarchy these tensors activate the tensor representation for the triangle.

Just as you are able to look at this curved triangle and understand that the difference in this triangle and the general concept of a triangle is that this triangle has curved line, this model will have the same intuition.

The triangle tensor that is activated has a lower activation than the original triangle tensor. This is because from the definition of the activation of a tensor, the part of the sum that encapsulates the type or neighborhood will have a lower activation. In this case the type of the tensor instead of being the line tensor is the curved tensor.

This type of representation has great properties that are not present in current architectures. Generalization. With this approach, even though the model has not been trained with pictures of curvy triangles, because it will learn that curves are similar to straight lines and place these two tensor representations in a relatively similar neighborhood

Explainable AI. I can explain that the reason that the figure is not 100% a triangle is because it is made up of curvy lines instead of straight lines. This information is stored in the ”neighborhood” property of the tensor.

4 Conclusion

Test set results in progress... will update soon.

References

https://arxiv.org/pdf/1411.6369.pdf