Title: Knowledge distillation

URL: https://en.wikipedia.org/wiki/Knowledge_distillation

PageID: 62295363

Categories: Category:Deep learning

In machine learning , knowledge distillation or model distillation is the process of transferring knowledge from a large model to a smaller one. While large models (such as very deep neural networks or ensembles of many models) have more knowledge capacity than small models, this capacity might not be fully utilized. It can be just as computationally expensive to evaluate a model even if it utilizes little of its knowledge capacity. Knowledge distillation transfers knowledge from a large model to a smaller one without loss of validity . As smaller models are less expensive to evaluate, they can be deployed on less powerful hardware (such as a mobile device). [1]

A less common technique called Reverse Knowledge Distillation transfers knowledge from a smaller model to a larger one. [2]

Model distillation is not to be confused with model compression , which describes methods to decrease the size of a large model itself, without training a new model. Model compression generally preserves the architecture and the nominal parameter count of the model, while decreasing the bits-per-parameter.

Knowledge distillation has been successfully used in several applications of machine learning such as object detection , [3] acoustic models , [4] and natural language processing . [5] Recently [when?] , it has also been introduced to graph neural networks applicable to non-grid data. [6]

Methods

Knowledge transfer from a large model to a small one somehow needs to teach the latter without loss of validity. If both models are trained on the same data, the smaller model may have insufficient capacity to learn a concise knowledge representation compared to the large model. However, some information about a concise knowledge representation is encoded in the pseudolikelihoods assigned to its output: when a model correctly predicts a class, it assigns a large value to the output variable corresponding to such class, and smaller values to the other output variables. The distribution of values among the outputs for a record provides information on how the large model represents knowledge. Therefore, the goal of economical deployment of a valid model can be achieved by training only the large model on the data, exploiting its better ability to learn concise knowledge representations, and then distilling such knowledge into the smaller model, by training it to learn the soft output of the large model. [1]

Mathematical formulation

Given a large model as a function of the vector variable x {\displaystyle \mathbf {x} } , trained for a specific classification task, typically the final layer of classification networks is a softmax in the form

where t {\displaystyle t} is the temperature , a parameter which is set to 1 for a standard softmax. The softmax operator converts the logit values z i (x) {\displaystyle z_{i}(\mathbf {x})} to pseudo-probabilities: higher temperature values generate softer distributions of pseudo-probabilities among the output classes. Knowledge distillation consists of training a smaller network, called the distilled model , on a data set called the transfer set (which is different than the data set used to train the large model) using cross-entropy as the loss function between the output of the distilled model y (x | t) {\displaystyle \mathbf {y} (\mathbf {x} |t)} and the output of the large model y ^ (x | t) {\displaystyle {\hat {\mathbf {y} }}(\mathbf {x} |t)} on the same record (or the average of the individual outputs, if the large model is an ensemble), using a high value of softmax temperature t {\displaystyle t} for both models [1]

In this context, a high temperature increases the entropy of the output, therefore providing more information to learn for the distilled model compared to hard targets, and at the same time reducing the variance of the gradient between different records, thus allowing a higher learning rate . [1]

If ground truth is available for the transfer set, the process can be strengthened by adding to the loss the cross-entropy between the output y i (x | 1) {\displaystyle y_{i}(\mathbf {x} |1)} of the distilled model computed with t = 1 {\displaystyle t=1} , and the known label y ¯ i {\displaystyle {\bar {y}}_{i}}

where the component of the loss with respect to the large model is weighted by a factor of t 2 {\displaystyle t^{2}} since, as the temperature increases, the gradient of the loss with respect to the model weights scales by a factor of 1 t 2 {\displaystyle {\frac {1}{t^{2}}}} . [1]

Relationship with model compression

Under the assumption that the logits have zero mean , it is possible to show that model compression is a special case of knowledge distillation. The gradient of the knowledge distillation loss E {\displaystyle E} with respect to the logit of the distilled model z i {\displaystyle z_{i}} is given by

where z ^ i {\displaystyle {\hat {z}}_{i}} are the logits of the large model. For large values of t {\displaystyle t} this can be approximated as

and under the zero-mean hypothesis ∑ j z j = ∑ j z ^ j = 0 {\displaystyle \sum _{j}z_{j}=\sum _{j}{\hat {z}}_{j}=0} it becomes z i – z ^ i N T 2 {\displaystyle {\frac {z_{i}-{\hat {z}}_{i}}{NT^{2}}}} , which is the derivative of 1 2 (z i – z ^ i) 2 {\displaystyle {\frac {1}{2}}\left(z_{i}-{\hat {z}}_{i}\right)^{2}} , i.e. the loss is equivalent to matching the logits of the two models, as done in model compression. [1]

"Optimal Brain Damage" algorithm

The Optimal Brain Damage (OBD) algorithm is as follows: [7]

Deleting a parameter means fixing the parameter to zero. The "saliency" of a parameter θ {\displaystyle \theta } is defined as 1 2 (∂ θ 2 L) θ 2 {\displaystyle {\frac {1}{2}}(\partial _{\theta }^{2}L)\theta ^{2}} , where L {\displaystyle L} is the loss function. The second-derivative ∂ θ 2 L {\displaystyle \partial _{\theta }^{2}L} can be computed by second-order backpropagation .

The idea for optimal brain damage is to approximate the loss function in a neighborhood of optimal parameter θ ∗ {\displaystyle \theta ^{*}} by Taylor expansion : L (θ) ≈ L (θ ∗) + 1 2 ∑ i (∂ θ i 2 L (θ ∗)) (θ i – θ i ∗) 2 {\displaystyle L(\theta)\approx L(\theta ^{*})+{\frac {1}{2}}\sum _{i}(\partial _{\theta _{i}}^{2}L(\theta ^{*}))(\theta _{i}-\theta _{i}^{*})^{2}} where ∇ L (θ ∗) ≈ 0 {\displaystyle \nabla L(\theta ^{*})\approx 0} , since θ ∗ {\displaystyle \theta ^{*}} is optimal, and the cross-derivatives ∂ θ i ∂ θ j L {\displaystyle \partial _{\theta _{i}}\partial _{\theta _{j}}L} are neglected to save compute. Thus, the saliency of a parameter approximates the increase in loss if that parameter is deleted.

History

A related methodology was model compression or pruning , where a trained network is reduced in size. This was first done in 1965 by Alexey Ivakhnenko and Valentin Lapa in USSR (1965). [8] [9] [10] Their deep networks were trained layer by layer through regression analysis. Superfluous hidden units were pruned using a separate validation set. [11] Other neural network compression methods include Biased Weight Decay [12] and Optimal Brain Damage. [7]

An early example of neural network distillation was published by Jürgen Schmidhuber in 1991, in the field of recurrent neural networks (RNNs). The problem was sequence prediction for long sequences, i.e., deep learning . It was solved by two RNNs. One of them (the automatizer) predicted the sequence, and another (the chunker) predicted the errors of the automatizer. Simultaneously, the automatizer predicted the internal states of the chunker. After the automatizer manages to predict the chunker's internal states well, it would start fixing the errors, and soon the chunker is obsoleted, leaving just one RNN in the end. [13] [14]

The idea of using the output of one neural network to train another neural network was also studied as the teacher-student network configuration. [15] In 1992, several papers studied the statistical

mechanics of teacher-student configurations with committee machines [16] [17] or both are parity machines. [18]

Compressing the knowledge of multiple models into a single neural network was called model compression in 2006: compression was achieved by training a smaller model on large amounts of pseudo-data labelled by a higher-performing ensemble, optimizing to match the logit of the compressed model to the logit of the ensemble. [19] The knowledge distillation preprint of Geoffrey Hinton et al. (2015) [1] formulated the concept and showed some results achieved in the task of image classification .

Knowledge distillation is also related to the concept of behavioral cloning discussed by Faraz Torabi et. al. [20]

References

External links

Distilling the knowledge in a neural network – Google AI