Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping

My method and Suggestions: - Task 4

Satyam Kumar (B19BB060)

Suggestions

They are using limited problems in this paper. So, it may be possible that the results are biased. Also, they need to specify which type of overfitting they are encountering. There are different methods of handling according to different types of overfitting. There are 3 types of overfitting: -

1.) **Noise learning on the training set**: when the training set is too small in size, or has fewer representative data or too many noises. This situation makes the noises have great chances to be learned, and later act as a basis of predictions. So, a well-functioning algorithm should be able to distinguish representative data from noises.

2.) **Hypothesis complexity**: the trade-off in complexity, a key concept in statistic and machining learning, is a compromise between Variance and Bias. It refers to a balance between accuracy and consistency. When the algorithms have too many hypotheses (too many inputs), the model becomes more accurate on average with lower consistency.

3.) **Multiple comparisons procedures** which are ubiquitous in induction algorithms, as well as in other Artificial Intelligence (AI) algorithms.

Along with that, in this paper they are using Early – stopping, it is used to avoid phenomenon "learning speed slow-down". So, there is an issue in this that the accuracy of algorithms stops improving after some point, or even getting worse because of noise learning. So, there is a technical update to solve this problem.

For neural nets, the learning process is to find a perfect set of weights and bias. The neurons learn at a rate of determined by the partial derivatives of the cost-function $\partial \boldsymbol{C}/\ \partial \boldsymbol{W}$ and $\partial \boldsymbol{C}/\ \partial \boldsymbol{b}$. So, the speed of learning depends on the values of these two partial derivatives.

Practically, to find out the point to stop learning, the obvious way is to keep track of accuracy on the test data as our network trains. In other words we compute the accuracy at the end of each epoch and stop training when the accuracy on test data stops improving.

1.) Network reduction

Noise reduction becomes one researching direction for overfitting inhibition. Pruning is proposed to reduce the size of finial classifiers in relational learning, especially in decision tree learning.

<u>Pre-pruning and post-pruning are two standard approaches used to dealing with noise: -</u>

Pre-pruning algorithms use stopping criteria to determine when to stop adding conditions to a rule, or adding rule to a model description, such as encoding length restriction based on the evaluation of encoding cost.

Post-pruning splits the training set into two subsets: growing set and pruning set. Compared to pre-learning, post-pruning algorithms ignore overfitting problems during the learning process on growing set. Instead, they prevent overfitting through deleting conditions and rules from the model generated during learning. This approach is much more accurate, and however, less efficient.

2.) Regularization

In this paper, they didn't do any work on equation for regularization. So, here I'm going to minimize the weights of the features which have little influence on the final classification. In other words, I am going to limit the effect of those useless features. However, I don't know which features are useless, so I try to limit them all by minimizing the cost function of given model in the paper. To do this, we add a "penalty term", called *regularizor*, to the cost function as shown in the following formula:

$$\mathrm{J}`(\omega;\ \mathrm{X},\mathrm{y}) = \mathrm{J}(\omega;\ \mathrm{X},\mathrm{y}) + \alpha\,\Omega(\omega)$$

$$J`(\omega; X, y) = 1/2\mathrm{m}||X_{\mathrm{w}} - y||_2^{\,2} + \alpha\,\Omega(\omega)$$

Where J(ω; X, y) is the original cost function ω is the weight, X is the training set, y is the labelled value, m is the size of training set, α is the regularization coefficient, and αΩ(ω) is the penalty term.

3.) L2 regularization (Weight Decay)

L2 regularization uses the "Ridge Regression" theory. In this approach, we use the Euclidean distance as the penalty term.

$$\Omega(\omega) = ||w||_2 = \sqrt{\sum_i w_i^2}$$

This approach makes the networks prefers to learn features with small weights. Instead of rejecting those less valuable features, we give them lower weights. So, we get as much information as possible. Large weights can only be given to the features that considerably improve the initial cost function. To find out the point to stop learning, the obvious way is to keep track of accuracy on the test data as our network trains. In another word, we compute the accuracy at the end of each epoch and stop training when the accuracy on test data stops improving.

4.) Dropout

Dropout is a popular and effective technique against overfitting in neural networks. The initial idea of dropout is to randomly drop units and relevant connections from the neural networks during training. This prevents units from co-adapting too much. The general process is as follow: -

- Drop randomly half of the hidden neurons to construct a new simpler network.
- Train the thinned network using stochastic gradient descent (The gradients for each parameter are averaged over the training cases in each mini-batch. Any training case which does not use a parameter contributes a gradient of zero for that parameter)
- Restore the removed neurons
- Randomly remove half of hidden neurons from the restored network to form a new thinned network
- Repeat the process above until get an ideal set of parameters

These are some methods which can be used in to reduce overfitting along with the one discussed in the paper like Early stopping along with backpropagation and Conjugate gradient.

One thing also to note that, these are the experimental results based on real life problems and also contains very large networks. So, it is difficult to train such model in our local machine and try to figure out results. So, all the above work is based on assumptions only.