0.1 Serial learning

In addition to parallel training a neural network on multiple tasks it is also possible to train a network in a serial order. Instead of training the parameters on the inputs from multiple tasks simultaneously, the network is first trained on task A and then on task B and so on. This presents quite a few challenges. For example compared to parallel training where during the training the network automatically searches for a common distribution of the parameters: θ. In serial learning after training on task A the network will have a distribution θ_A, but when it starts to train on task B it changes the parameters to a distribution θ_B which might have nothing in common with θ_A. The network will then perform really badly on the first task and if it would be trained on another task C it would perform badly on both task A and task B.

This is called *Catastrophic Forgetting.* Where unlike the human brain, which is able to learn similar tasks without forgetting the previous ones, the neural network forgets the previous task. One could even expect the contrary to happen: knowing a taks that is similar to a new one should be able to help you learn it. A new interesting technique used to try to emulate this behavior is Elastic weight consolidation(EWC)[11].

0.1.1 Elastic weight consolidation

When a neural network is trained a set of weights and biases, θ, is adjusted with the goal of optimizing the performance. There exists quite a lot of variations on this set that won't affect the performance of the neural network too much. This is mostly cause by overparametrization. It's a reasonable assumption that if we have two tasks A and B that are very similar that there exists a set of weights and biases, θ_B, that lies close to a set optimized for task A , θ_A an is an optimized set for task B. Now of course the goal is to find this set.

EWC does this by anchoring the parameters to the set trained on task A. Then when the network is trained on task B they will try to satay close to the previous solution. The anchoring is done by pretending the parameters are stuck to their original values via a spring with a certain stiffness. It's this stifness where the important innovation of EWC comes from. Instead of having a constant stiffness for each parameter. It changes based on how important that parameter was to the previous task, the more important the parameter the stiffer the string.

The reasoning for this constraint comes from seeing a neural network from a probabilistic perspective. Where the objective is to find the most probable parameters θ for the given data D. We can calculate this posterior probability $p(\theta|D)$ from the prior probability $p(D|\theta)$ using bayes' rule.

$$\log p(\theta|D) = \log p(D|\theta) - \log p(D) + \log p(\theta) \tag{1}$$

The prior probability simpifies to the negative of the loss function $-\mathcal{L}(\theta)$. If we now split the dat in two parts: one for task A and the other for task B we can write it as follows:

$$\log p(\theta|D) = \log p(D_B|\theta) - \log p(D_B) + \log p(\theta|D_A) \tag{2}$$

From this we can see that all the information from task A seems to have been absorbed by the posterior probability $\log p(\theta|D_A)$. The posterior probability itself can not be calculated but can be approximated. In ECW the posterior probability is approximated as a gaussian distribution with mean θ_A and a diagonal precision given by the diagonal of the fisher information matrix F. The loss now looks like this:

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta_{A,i})^2 \tag{3}$$

Here F_i os the value on the diagonal of the Fisher information matrix corresponding to parameter i. λ is a parameter that changes value based on how important task A is to task B. This is basically an L2 Norm with an added weight based on the importance of the parameters for the new task.

0.1.2 Fisher information

Fisher information is derived from a very general question: "How much information does the measurable data y from a system contain about an unknown parameter a on which it is dependent?". Since all we have to go on is the data y, an estimation function $\hat{a}(y)$ has to be formed, that has to property to be a correct estimation of a on average. This means $\langle \hat{a}(y) \rangle = a$ or equivalently [12, p. 9]:

$$\langle \hat{a}(y) \rangle = \int (\hat{a}(y) - a) p(y|a) dy = 0 \tag{4}$$

Estimators obeying 4 are called "unbiased estimators". To get to the Fisher information we start by differentiating this estimator to a.

$$\int (\hat{a}(y) - a) \frac{\partial p}{\partial a} dy - \int p dy = 0, p = p(y|a) \tag{5}$$

Using the identity $\frac{\partial p}{\partial a} = p \frac{\partial \ln(p)}{\partial a}$ and the normalization property $\int p dy = 1$

$$\int (\hat{a}(y) - a) p \frac{\partial \ln(p)}{\partial a} dy = 1 \tag{6}$$

Factoring the integrand gives us:

$$\int [(\hat{a}(y) - a)\sqrt{p}] \sqrt{p} \frac{\partial \ln(p)}{\partial a} dy = 1 \tag{7}$$

Now we can use the schwarz inequality by factoring 7

$$\int [(\hat{a}(y) - a)\sqrt{p}]^2 dy \int [\sqrt{p} \frac{\partial \ln(p)}{\partial a} dy]^2 \geq 1 \tag{8}$$

$$\int [(\hat{a}(y) - a)^2 p] dy \int [\sqrt{p} (\frac{\partial \ln(p)}{\partial a})^2 dy] \geq 1 \tag{9}$$

Both of these integrals is an estimation the first being the mean squared error:

$$\int [(\hat{a}(y) - a)^2 p] dy = \langle (\hat{a}(y) - a)^2 \rangle = e^2 \tag{10}$$

The second integral is the Fisher information:

$$\int [\sqrt{p} (\frac{\partial \ln(p)}{\partial a})^2 dy] = \langle (\frac{\partial \ln(p)}{\partial a})^2 \rangle = I(a) \tag{11}$$

Here we see that I measures the gradient value of $p(y|a)$, thus the slower p changes with a the lower $I(a)$ will be. This also means that if $p(y|a) = p(y)$, meaning y is independent of a then $I(a) = 0$. This intuitively makes a lot of sense: if the data is independent of a then it certainly contains no information on it. If we imagine that $p(y|a)$ changes slowly with a then the estimation $\hat{a}(y)$ will have difficulty distinguishing between those different a values. The error $\langle (\hat{a}(y) - a) \rangle^2 = e^2$ will be quite big then whilst $I(a)$ will be smaller. And if $p(y|a)$ changes more quickly with a the opposite will happen: e^2 will be smaller and $I(a)$ will be bigger. Leading us to conclude that y contains a lot of information about a. Which was the initial goal of deriving the Fisher information.

0.1.3 Fisher information for multiple parameters

The derivation above was for a single parameter a, but imagine we have a system from which we use N data $\boldsymbol{y} = y_0 ... y_N$ which depend on N parameters $\boldsymbol{a} = a_0 ... a_N$. Then instead of just defining the scalar Fisher information $I(a)$, we now have the fisher information matrix F [12, p. 13]:

$$F_m n = \langle \frac{\partial \ln(p)}{\partial a_m} \frac{\partial \ln(p)}{\partial a_n} \rangle = \int \frac{\partial \ln(p)}{\partial a_m} \frac{\partial \ln(p)}{\partial a_n} d\boldsymbol{y}, p = p(\boldsymbol{y}|\boldsymbol{a}) \tag{12}$$

The diagonal of the Fisher information matrix again simplifies to the form for a single parameter.

$$F_{mm} = \langle (\frac{\partial \ln(p)}{\partial a_m})^2 \rangle \tag{13}$$

References

[1] McCulloch, Warren; Walter Pitts, *"A Logical Calculus of Ideas Immanent in Nervous Activity"*, 1943.

[2] Alex Graves *Generating sequences with recurrent neural networks*,2014

[3] A. Graves *Supervised sequence labeling with recurrent neural networks*,

[4] Ghoshal, A, Swietojanski, P & Renais, S, *Multilingual training of deep neural networks*, 2013

[5] Schultz, T & Waibel, A, *Language-independent and language-adaptive acoustic modeling for speech recognition*, 2001

[6] Heigold,G;Vanhoucke, V;Senior, A;Nguyen, P;Ranzato, M;Devin, M; Dean, J; *Multilingual acoustic models using distributed deep neural networks*

[7] Chan, W;Jaitly, N;Le, Quoc V.;Vinyals, O *Listen, Attend and Spell*

[8] Christopher M. Bishop *Neural Networks for pattern recognition*

[9] A. C. C Coolen, R. Kuhn, P. Sollich *Theory of Neural Information Processing Systems*

[10] Rich Caruana, *multitask learning*

[11] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, Raia Hadsell *Overcoming catastrophic forgetting in neural networks*,2017

[12] B. Roy Frieden, Robert A. Gatenby *Exploratory Data analysis using Fisher information*,2007