11-785 21/05/2020

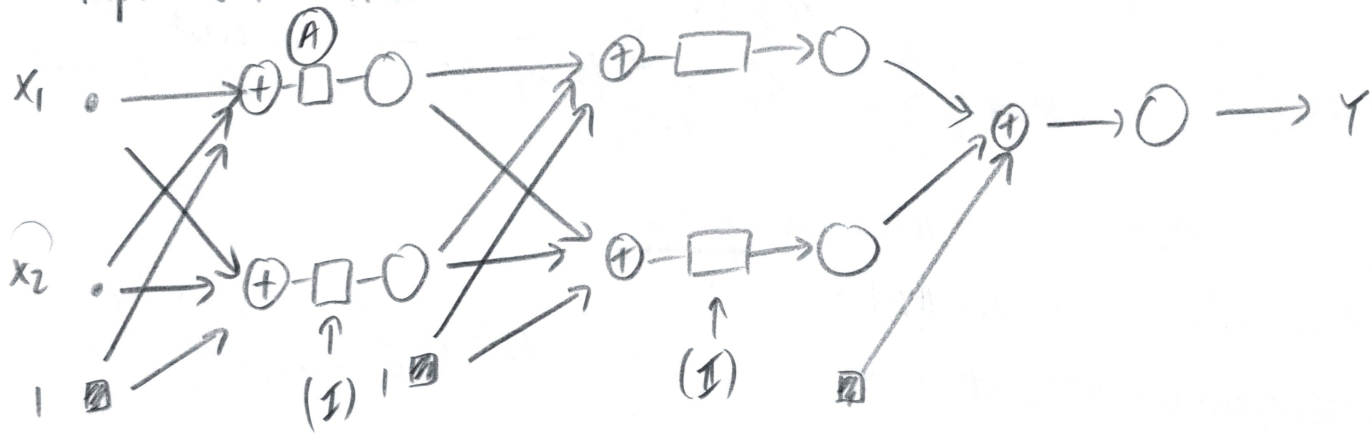
L7 - SGD, Overfitting, regularisation (BN part)

- Batch Normalisation (part of HW1)

- Diagram

- 2 hidden layer MLP

- BN - covariate adjust. unit that happens after the weighted addition of inputs (affine transformations), and before the activations.



- Note BN related transforms at (I) and (II).

Individual/close-up view

- BN happens at an individual unit
- Training → adjustment occurs over individual mini-batches
- BN aggregates stats over a mini-batch at a specific neuron, ~~normalises~~

1) Every instance within the mini-batch at the particular unit will end up being normalised by statistics computed over the mini-batch

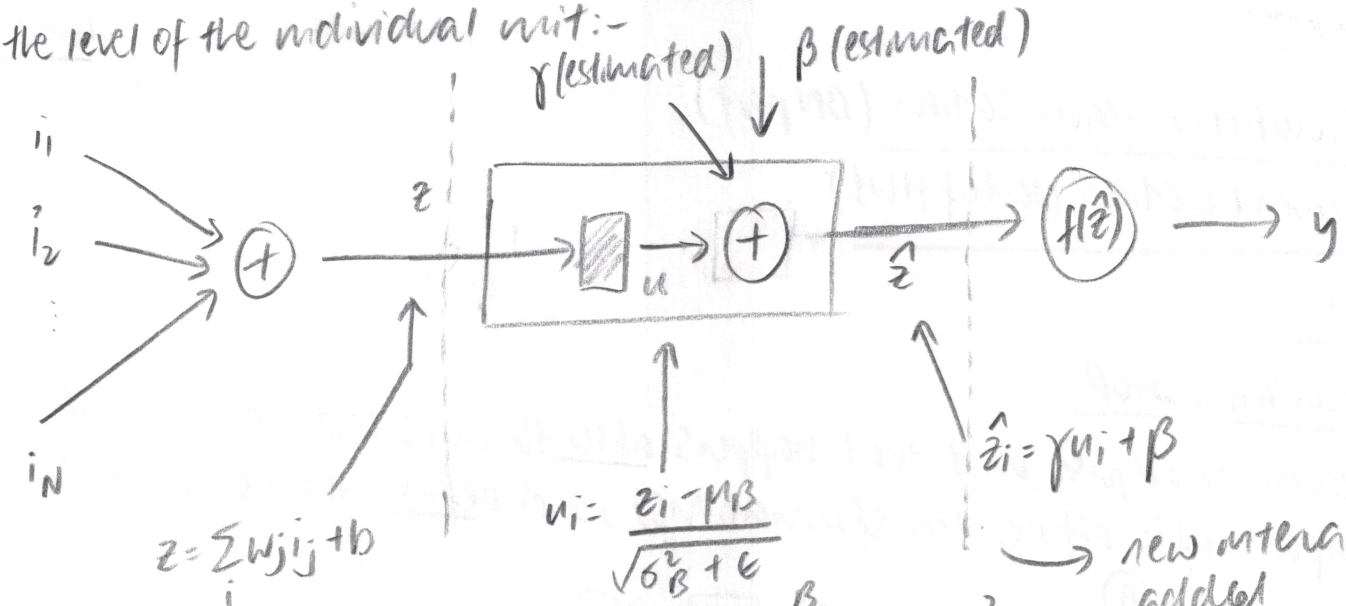
2) A further linear transformation is applied on every ~~instance~~ normalised instance within the mini-batch.

(to a unit-specific location)

(a)

- See supplementary notes on Batch Norm; and Ioffe and Szegedy (2015)

- At the level of the individual unit:-



- Minibatch size B:- $\mu_B = \frac{1}{B}\sum_{i=1}^{B} z_i \quad \sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(z_i - \mu_B)^2$

(*) Next piece of notation in lectures is confusing

(*) Want to refresh memory that:-

(I) i) Batch gradient descent:- N-size of training set

$$W_R \leftarrow W_R - \eta\left(\frac{1}{N}\sum_{t=1}^{N} \nabla_{W_R} Div(Y_t, d_t)\right)$$

ii) Incremental update (with SGD)

$$W_R \leftarrow W_R - \eta\left(\nabla_{W_R} Div(Y_t, d_t)\right)$$

B/m-size of mini-batch

iii) Mini-batch gd.

$$W_R \leftarrow W_R - \eta\left(\frac{1}{m}\sum_{t=1}^{m} \nabla_{W_R} Div(Y_t, d_t)\right)$$

(*) (P/S1) - note mini-batches is analogous to statistical sampling

↳ more details when clarified

(II) under batchnormalisation; the function $Div(Y_t, d_t)$ takes additional arguments μ_B and σ_B^2

i.e. we have $\left(\frac{1}{m}\sum_{t=1}^{m} \nabla_{W_R} Div(Y_t(X_t, \mu_B, \sigma_B^2), d_t\right)$

(*) Lecture notes state that $d_t = d_t(X_t)$; but I'm not sure how the target label d_t can be a function of input.

Typo?

(!!) Key point

- μ_B and σ_B^2 are functions of all the other training instances within the mini-batch (they are statistics)

- i.e. $\mu_B = f(x_1, \ldots, x_m)$ and $\sigma_B^2 = g(x_1, \ldots, x_m, \mu_B)$

- Rendering:-

$$\frac{1}{m}\sum_{t=1}^{m} \nabla_{w,R} Div\left(y_t(x_t, \mu_B(x_1, \ldots, x_m), \sigma_B^2(x_1, \ldots, x_m, \mu_B)), d_t\right)$$

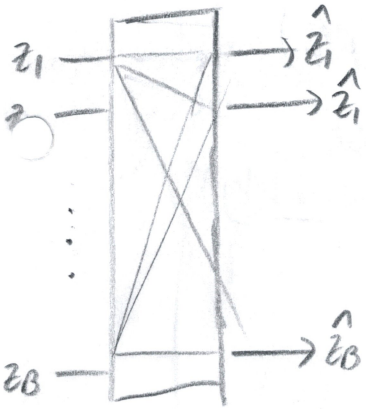
(*) this complicates backpropagation; but not to the degree of intractability

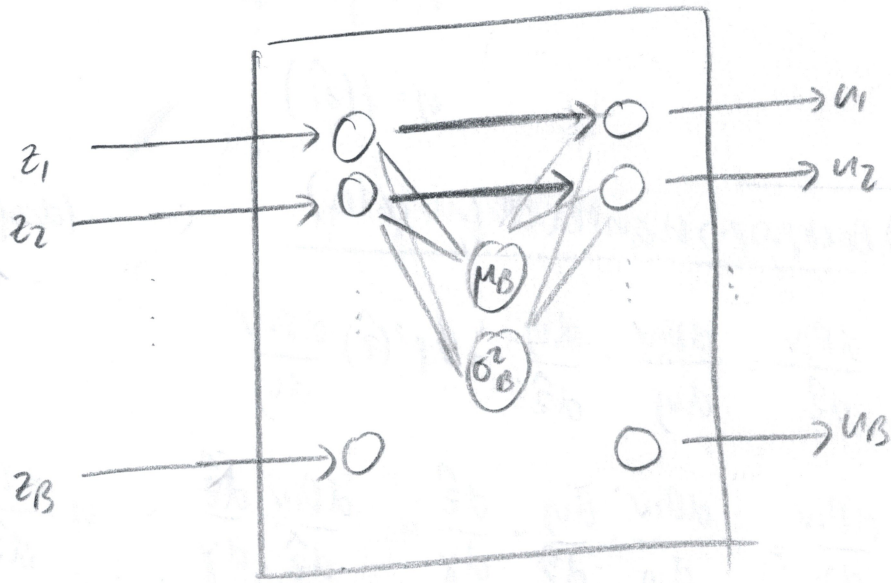
(*) Batchnorm as a vector function over a MINIBATCH

(*) Note it is not a vector function over dimensions (as we saw with softmax)

- It is a vector function over all inputs from a mini-batch
- computing $\frac{\partial Div}{\partial z_i} \Rightarrow$ consider all $\hat{z}_j$s

some influence diagrams:-



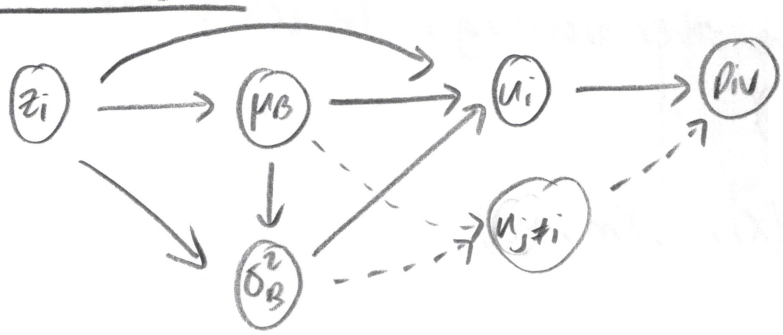


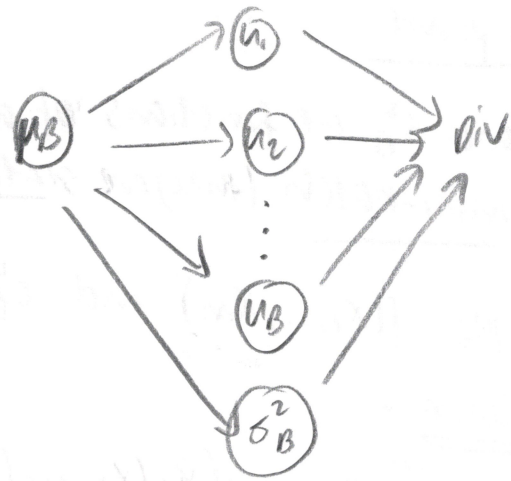
(*) Inputs/outputs are different instances in a minibatch

(*) BN occurring at level of individual neuron

(*) use these to aid understanding; vector function deriv to be assisted with next diagram

Influence diagrams





(*) BN - backpropagation

- start at end; move backwards recursively; forward first (for one instance within batch), end at a div. neuron

NO BN

$$z_i = \sum_j w_{ji} i_j + b$$

$$y_i = f(z_i)$$

with BN

$$z_i = \sum_j w_{ji} i_j + b$$

$$u_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

$$\mu_B = \frac{1}{B}\sum_{i=1}^{B} z_i \qquad \sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(z_i - \mu_B)^2$$

$$\hat{z}_i = \gamma u_i + \beta$$

$$y = f(\hat{z}_i)$$

(*) Backprop → high level (no calc.)

(drop all subscripts)
- to preserve emphasis on generality

- $\frac{dDiv}{d\hat{z}} = \frac{dDiv}{dy} \cdot \frac{dy}{d\hat{z}} = f'(\hat{z}) \frac{dDiv}{dy}$

- $\frac{dDiv}{d\gamma} = \underbrace{\frac{dDiv}{dy} \cdot \frac{dy}{d\hat{z}}}_{\text{full calc.}} \cdot \frac{d\hat{z}}{d\gamma} = \underbrace{\frac{dDiv}{d\hat{z}}}_{\text{recursive part}} \underbrace{\frac{d\hat{z}}{d\gamma}}_{=u} = u \frac{dDiv}{d\hat{z}}$

- $\frac{dDiv}{d\beta} = \frac{dDiv}{d\hat{z}} \cdot \underbrace{\frac{d\hat{z}}{d\beta}}_{=1} = \frac{dDiv}{d\hat{z}}$

- $\frac{dDiv}{du} = \frac{dDiv}{d\hat{z}} \cdot \frac{d\hat{z}}{du} = \gamma \frac{dDiv}{d\hat{z}}$

- we have:-

$$\frac{\partial Div}{\partial \mu_B} = \sum_{i=1}^{B} \frac{\partial Div}{\partial u_i}\frac{\partial u_i}{\partial \mu_B} + \frac{\partial Div}{\partial \sigma_B^2}\cdot\frac{\partial \sigma_B^2}{\partial \mu_B}, \quad u_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2+\epsilon}} \quad \sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B} z_i^2 - 2z_i\mu_B + \mu_B^2$$

(a) (b)

From (a):- $\frac{\partial u_i}{\partial \mu_B} = \frac{-1}{\sqrt{\sigma_B^2+\epsilon}}$

From (b):- $\frac{\partial \sigma_B^2}{\partial \mu_B} = \frac{\partial}{\partial \mu_B}\frac{1}{B}\sum_{i=1}^{B} z_i^2 - 2z_i\mu_B + \mu_B^2$

$$= \frac{1}{B}\sum_{i=1}^{B}\frac{\partial}{\partial \mu_B}\left(z_i^2 - 2z_i\mu_B + \mu_B^2\right) = \frac{1}{B}\sum_{i=1}^{B} -2z_i + 2\mu_B$$

$$\Rightarrow \frac{\partial \sigma_B^2}{\partial \mu_B} = \frac{\sum_{i=1}^{B} -2(z_i - \mu_B)}{B}$$

$$\Rightarrow \frac{\partial Div}{\partial \mu_B} = \sum_{i=1}^{B}\frac{\partial Div}{\partial u_i}\left(\frac{-1}{\sqrt{\sigma_B^2+\epsilon}}\right) + \underbrace{\frac{\partial Div}{\partial \sigma_B^2}\left(\frac{\sum_{i=1}^{B} -2(z_i-\mu_B)}{B}\right)}_{=0}$$

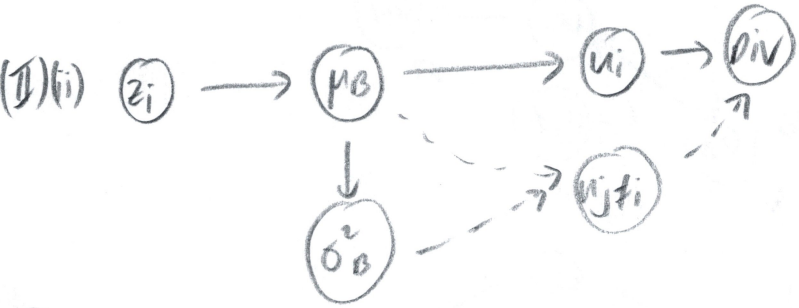
(more explicitly)

- Note: $\frac{\sum_{i=1}^{B} -2(z_i-\mu_B)}{B} = -2\sum_{i=1}^{B}\frac{z_i-\mu_B}{B} = -2\left\{\sum_{i=1}^{B}\frac{z_i}{B} - \sum_{i=1}^{B}\frac{\mu_B}{B}\right\} = -2\{\mu_B - \mu_B\}$

(***)

Hence: $\frac{\partial Div}{\partial \mu_B} = \sum_{i=1}^{B}\frac{\partial Div}{\partial u_i}\left(\frac{-1}{\sqrt{\sigma_B^2+\epsilon}}\right)$

(II)(i)



(ii)

$$\frac{\partial \mu_B}{\partial z_i} = \frac{1}{B}$$

- And more tricky, but not intractably (reinserting subscripts)

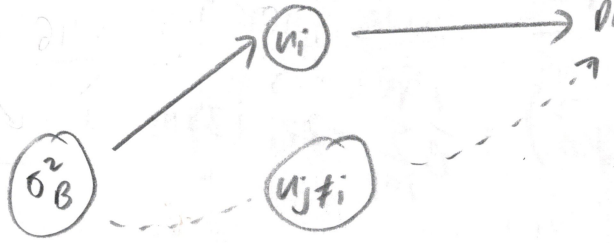
- $div = f(u_i, \mu_B, \sigma_B^2)$ ⟶ (*) MUST READ THIS WITH INFLUENCE DIAG. (*)

- $\frac{\partial div}{\partial z_i} = \frac{\partial div}{\partial u_i} \cdot \frac{\partial u_i}{\partial z_i} + \frac{\partial div}{\partial \sigma_B^2} \cdot \frac{\partial \sigma_B^2}{\partial z_i} + \frac{\partial div}{\partial \mu_B} \cdot \frac{\partial \mu_B}{\partial z_i}$

(II) (I) (II)

(*) use influence diag to evaluate each term:-

(I)(i)



(i) $u_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$

- $\frac{\partial div}{\partial z_i} = \sum_{j=1}^{B-1} \frac{\partial div}{\partial u_j} \frac{\partial u_j}{\partial \sigma_B^2} + \frac{\partial div}{\partial u_i} \frac{\partial u_i}{\partial \sigma_B^2} = \sum_{i=1}^{B} \frac{\partial div}{\partial u_i} \frac{\partial u_i}{\partial \sigma_B^2}$

indirect direct

- From (i); $\frac{\partial u_i}{\partial \sigma_B^2} = -\frac{1}{2}(z_i - \mu_B)(\sigma_B^2 + \epsilon)^{-3/2}$

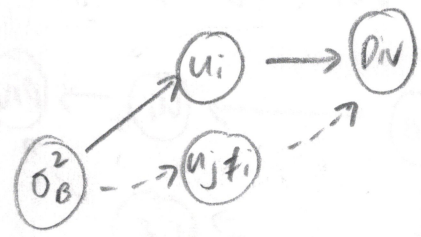
- Hence $\frac{\partial div}{\partial \sigma_B^2} = \sum_{i=1}^{B} \frac{\partial div}{\partial u_i} \frac{\partial u_i}{\partial \sigma_B^2} = \sum_{i=1}^{B} \frac{\partial div}{\partial u_i} \cdot -\frac{1}{2}(z_i - \mu_B)(\sigma_B^2 + \epsilon)^{-3/2}$

(*) $\rightarrow \frac{\partial div}{\partial \sigma_B^2} = -\frac{1}{2}(\sigma_B^2 + \epsilon)^{-3/2} \sum_{i=1}^{B} \frac{\partial div}{\partial u_i}(z_i - \mu_B)$

(I)(ii)

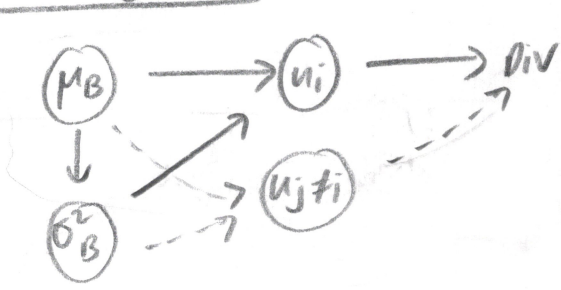
(*) $\frac{\partial \sigma_B^2}{\partial z_i} = \frac{2(z_i - \mu_B)}{B}$

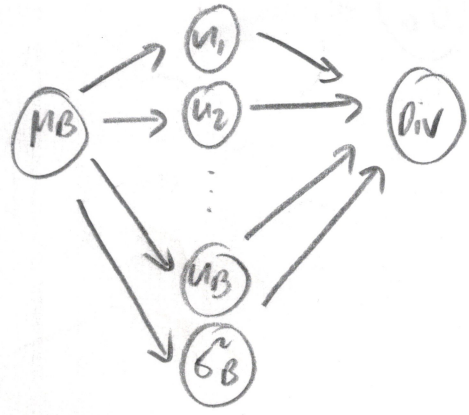
(ii)



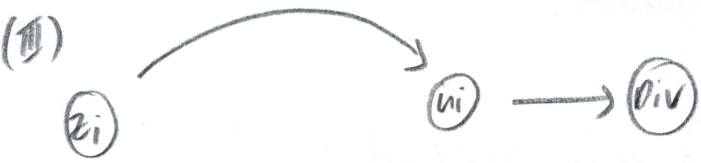
(II) Influence diagram (trajec.)

(i)





(III)



$$u_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

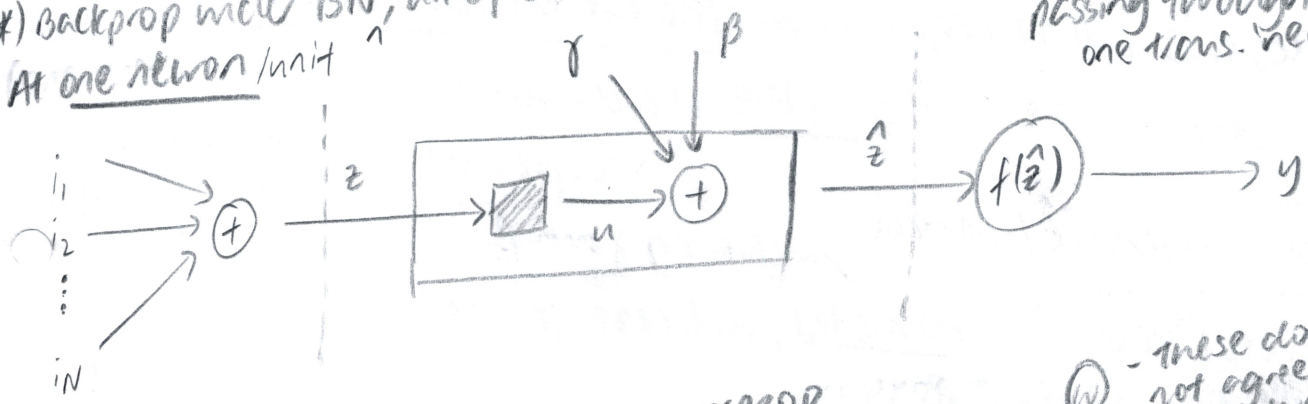
- From above:- (✱)

$$\frac{\partial Div}{\partial u_i} \cdot \frac{\partial u_i}{\partial z_i} = \frac{\partial Div}{\partial u_i} \left(\frac{1}{\sqrt{\sigma_B^2 + \epsilon}} \right)$$

(III) This only shows BN transformations applied to one training instance passing through one trans. neuron.

(✱) Backprop with BN, all eq. summarised:-

- At one neuron/unit



FORWARD

$$z_i = \sum_j w_j i_j + b$$

$$\mu_B = \frac{1}{B} \sum_{i=1}^{B} z_i$$

$$\sigma_B^2 = \frac{1}{B} \sum_{i=1}^{B} (z_i - \mu_B)^2$$

$$u_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

$$\hat{z}_i = \gamma u_i + \beta$$

$$y = f(\hat{z}_i)$$

BACKPROP

$$\frac{dDiv}{d\hat{z}_i} = f'(\hat{z}) \frac{dDiv}{dy_i}$$

$$\frac{dDiv}{d\gamma} = u_i \frac{dDiv}{d\hat{z}_i}$$

$$\frac{dDiv}{d\beta} = \frac{dDiv}{d\hat{z}_i}$$

$$\frac{dDiv}{du_i} = \gamma \frac{dDiv}{d\hat{z}_i}$$

(IV) - these do not agree with HW1 deriv; nor Ioffe & Szegedy (2015)

$$\frac{\partial Div}{\partial z_i} = \frac{\partial Div}{\partial u_i} \frac{\partial u_i}{\partial z_i} + \frac{\partial Div}{\partial \sigma_B^2} \frac{\partial \sigma_B^2}{\partial z_i} + \frac{\partial Div}{\partial \mu_B} \frac{\partial \mu_B}{\partial z_i}$$

$$\frac{\partial Div}{\partial z_i} = \frac{\partial Div}{\partial u_i} \left(\frac{1}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \frac{\partial Div}{\partial \sigma_B^2} \left(\frac{2(z_i - \mu_B)}{B} \right) + \frac{\partial Div}{\partial \mu_B} \cdot \frac{1}{B}$$

where $\frac{\partial Div}{\partial \sigma_B^2} = -\frac{1}{2} (\sigma_B^2 + \epsilon)^{-3/2} \sum_{i=1}^{B} \frac{\partial Div}{\partial u_i} (z_i - \mu_B)$; $\frac{\partial Div}{\partial \mu_B} = \frac{-1}{\sqrt{\sigma_B^2 + \epsilon}} \sum_{i=1}^{B} \frac{\partial Div}{\partial u_i}$

(*) Backprop continues as usual from $\frac{\partial Div}{\partial z_i}$ onwards.

(W): For HW1; and completeness,
it is possible to write the calculations in a more general form
- see supplementary notes with HW.
- using a vector notation? (although at time of writing; it is unclear why vector notation is used with summations rather than dot products; as in why not economise given more abstract rep)
- in any case; this will be required for efficient implementation in Numpy (*) And above is not complete as a derivation ... (!!)
→ supplement → see HW1 as well for implement details.

(*) Batch Norm - inference / test stage, including γ and β

- After parameters have been estimated; we freeze 'parameters'
- During test time; we require

$$\mu_{BN} = \frac{1}{N_B}\sum_{b=1}^{N_B} \mu_{B,b} \quad \text{where } N_B \text{ - no. of batches}$$

$$\sigma^2_{BN} = \frac{B}{(B-1)} \cdot \frac{1}{N_B}\sum_{b=1}^{N_B} \sigma^2_{B,b}$$

(*) The key idea is that for an individual activation;
we will have multiple computations of mini-batch means and variances μ_B and σ^2_B.
- More explicitly; if we have $1, 2, 3, \ldots, N_B$ mini-batches (i.e. partitions of the training set)
- we will have computed $(\mu_{B,1}, \sigma^2_{B,1})$ for batch 1

$(\mu_{B,2}, \sigma^2_{B,2})$ — " — 2

$\vdots$

$(\mu_{B,N_B}, \sigma^2_{B,N_B})$ — " — N_B.

- And that is for 1 activation unit

(*) some statistical reinforcement to link with Ioffe & Szegedy

- each mini-batch is a random sample of the entire training set (population); a mini-batch created by drawing B instances from training data.
- as data-points comprising a mini-batch are randomly sampled, mini-batch statistics are functions of randomly selected training instances (albeit deterministic)
- as mini-batch statistics can also be viewed as random variables; and as estimators of the mean and variance of each activation
  - Note $z_1, z_2, \ldots, z_N$ - IID r.v.s.

$$\mu_B = \frac{1}{B}\sum_{i=1}^{B} z_i \qquad \sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(z_i - \mu_B)^2$$

(*) to further clarify;
we have a distri of the data f_z
distri of statistic f_{μ_B} and $f_{\sigma_B^2}$

(*) μ_B and σ_B^2 as mini-batch mean and variance are a little misleading from a stats pespective, notationally should really be sample mean $\mu_B = \bar{z}_B$. σ_B^2 is actually an estimator of the variance of that activation, with no bias adjustment (biased sample variance)

(*) during inference, we use population statistics rather than mini-batch statistics.

- so we use the mini-batch sample means and sample variances to get our population statistics:-
  (or an estimate of them if implementation leads to that)

$$\mathbb{E}[z_i] = \mu_{BN} = \frac{1}{N_B}\sum_{b=1}^{N_B} \mu_{B,b} = \mathbb{E}_{\mu_B \sim p(\mu_B)}[\mu_B]$$

$$\text{Var}[z_i] = \sigma_{BN}^2 = \frac{B}{B-1}\cdot\frac{1}{N_B}\sum_{b=1}^{N_B}\sigma_{B,b}^2 = \frac{B}{B-1}\mathbb{E}_{\sigma_B^2 \sim p(\sigma_B^2)}[\sigma_B^2]$$

(unbiased variance estimate)

(*) Note:-

$$\mathbb{E}[z_i] = \frac{1}{N_B}\sum_{b=1}^{N_B} \mu_{B,b} = \frac{1}{N_B}\left(\mu_{B,1} + \mu_{B,2} + \ldots + \mu_{B,N_B}\right)$$

$$= \frac{1}{N_B}\left(\frac{1}{B}\sum_{i=1}^{B} z_i + \frac{1}{B}\sum_{j=1}^{B} z_j + \ldots + \frac{1}{B}\sum_{k=1}^{B} z_k\right)$$

$$= \frac{1}{N_B B}\sum_{i=1}^{N_B B} z_i = \frac{1}{N}\sum_{i=1}^{N} z_i$$

and similarly

$$\mathrm{Var}[z_i] = \frac{B}{B-1}\,\mathbb{E}_{\sigma_B^2}[\sigma_B^2] = \mathbb{E}\left[\frac{B}{B-1}\sigma_B^2\right] = \mathbb{E}[S_B^2]$$

→ relation between variance of activations under distr of data, and the mean of the distribution of the unbiased sampling variance of the activation.

-(*) implementation requires a running estimate being kept, and may prevent above from holding in some cases

(STI) (BK) - once training complete:- for each activation and training example (after param est.)

1) Normalisation

$$n_i = \frac{z_i - \mu_{BN}}{\sqrt{\sigma_{BN}^2 + \epsilon}}$$

(using population statistics NOT mini-batch stats)

2) Linear transform:-

$$\hat{z}_i = \gamma\left(\frac{z_i - \mu_{BN}}{\sqrt{\sigma_{BN}^2 + \epsilon}}\right) + \beta = \frac{\gamma}{\sqrt{\sigma_{BN}^2 + \epsilon}} z_i + \left(\beta - \frac{\gamma \mu_{BN}}{\sqrt{\sigma_{BN}^2 + \epsilon}}\right)$$

(Ioffe & Szegedy 2016)

(*) See pseudocode of original paper, and also some calcs / implmt details in HW1.

Batch Normalisation (Ioffe & Szegedy 2015)

19/05/2016

- $\Theta = \underset{\Theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell(x_i, \Theta)$

- $\{x_1, \ldots, x_N\}$ - training set
- $\{x_1, \ldots, x_m\}$ - mini batch

- DL: (for anchoring context) :-

$$J^*(\theta) = \sum_x \sum_y p_{data}(x, y)\, L(f(x;\theta), y)$$

$$g = \nabla_\theta J^*(\theta) = \sum_x \sum_y p_{data}(x, y)\, \nabla_\theta L(f(x;\theta), y)$$

$$\hat{g} = \frac{1}{m} \nabla_\theta \sum_i L(f(x^{(i)};\theta), y^{(i)})$$

- use mini-batch to approx. gradient of loss via :-

$$\frac{1}{m} \frac{\partial \ell(x_i;\Theta)}{\partial \Theta}$$

(*) mini batch:

i) gradient over loss over mini-batch is an estimate of loss gradient over entire training set.

ii) Parallel computation.

(*) Rationale/observation for BN

- $\ell = F_2(F_1(u, \Theta_1), \Theta_2)$ $\quad F_1, F_2$ - arbitrary trans.

- $\ell = F_2(x, \Theta_2)$ where $x = F_1(u, \Theta_1)$

- gradient descent :-

$$\Theta_2 \leftarrow \Theta_2 - \frac{\alpha}{m} \frac{\partial F_2(x_i, \Theta_2)}{\partial \Theta_2}$$

- batch size m
- learning rate α

is equivalent to standalone network F_2 with input x. (?) → not entirely clear. (*)

⇒ keep distn of x fixed ⇒ Θ_2 does not adjust to compensate for change in distn of x

(*) BN as altern. strategy for saturation, vanishing grad. → ~~spare time~~

(#) Fixed distri inputs to a subnet; consequences ~~inside~~ outside sub-net.

$z = g(\underline{W}\underline{u} + \underline{b})$ (sigmoid act.)

$\underline{u}$ - layer input $\underline{W}$ - weight matrix

$\underline{b}$ - bias vector $g(x) \frac{1}{1+\exp(-x)}$

- $|x| \to \infty$; $g'(x) \to 0$ ✓
- $\forall x_1, x_2, \ldots, x_K$ in $\underline{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_K \end{pmatrix} = \underline{W}\underline{u} + \underline{b}$ (?)

(#) All dim except those with small abs. values will train slowly

- $\underline{x}$ unaffected by $\underline{W}$, $\underline{b}$ and parmetes of layes below
- Δ to those param during training → "more many dimensions of $\underline{x}$" into the saturated regime of non-linearity, and slow convergence (?)
- Amplified as network depth ↑
- known as saturation problem ⇒ vanishing gradients

(*) Rectified with:-

i) ReLU (Nair & Hinton 2010)

ii) careful init (Bergio & Glorot 2010; Saxe et al. 2013)

iii) small learning rates

(#): Ensure that distri of nonlinearity inputs remains more stable as network trains then; optimise will not get stuck in sat. regime and training would accelerate

---

- Key ideas

(*) Internal covariate shift - chage in distri of internal nodes in network (network activations)

(*) Batch Normalisation - reduce covariate shift; accelerate DNN training

- ~~Normalisation~~

→

(*) Key claims

1) Beneficial effect on gradient flow
   ↳ Reduce dependence of gradients on scale of param / initial vals.

2) Allows use of higher learning rates without risk of divergence

3) Regularisation

4) Reduces need for dropout (Srivastava et al. 2014)

---

2. Discussion of trajectory → reducing ICS

---

3. Norm. via mini-batch stats

(*) Normalise each scalar feature map.
   - i.e. zero mean, unit variance — TRAIN SET

(*) Layer with d-dim input:-

$$\underline{x} = \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(d)} \end{pmatrix}$$

(*) Normalise each dim :-

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mathbb{E}[x^{(k)}]}{\sqrt{\mathrm{Var}[x^{(k)}]}} \qquad \forall\ k = 1, \ldots, d$$

$\mathbb{E}(\cdot)$; $\mathrm{var}(\cdot)$ over training set

(*) Introduce for each activation $x^{(k)}$, a pair of param. $\gamma^{(k)}$ and $\beta^{(k)}$ which scale and shift normalised value $\hat{x}^{(k)}$:-

$$y^{(k)} = \gamma^{(k)} \hat{x}^{(k)} + \beta^{(k)} \qquad \forall\ k = 1, \ldots, d$$

(*) Estimate $\gamma^{(k)}$ and $\beta^{(k)}$ ($\forall\ k = 1, \ldots, d$) with orig param.

(*) Restore rep. power of net ; can recover $x^{(k)}$ (original act.) with $\gamma^{(k)} = \sqrt{\mathrm{Var}[x^{(k)}]}$ and $\beta^{(k)} = \mathbb{E}[x^{(k)}]$

(*) Accept for mini-batches

(*) minibatch size m.

(*) focus on single activ./dimension $x^{(k)}$, drop k.

(*) m-values of activation in mini-batch

$$\mathcal{B} = \{x_1, x_2, \ldots, x_m\} = \{x_{1\ldots m}\}$$

(*) normalised values $\{\hat{x}_{1\ldots m}\}$

(*) linear trans $\{y_{1\ldots m}\}$

(*) BN trans: $BN_{\gamma,\beta}: x_{1\ldots m} \longrightarrow y_{1\ldots m}$

BN transform Algo (one activ./dim) NOT instance

IN: values of x over mini-batch $\mathcal{B} = \{x_{1\ldots m}\}$

OUT: $\{y_i = BN_{\gamma,\beta}(x_i)\}$ ϵ-num stab constat

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}$$

$$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$$

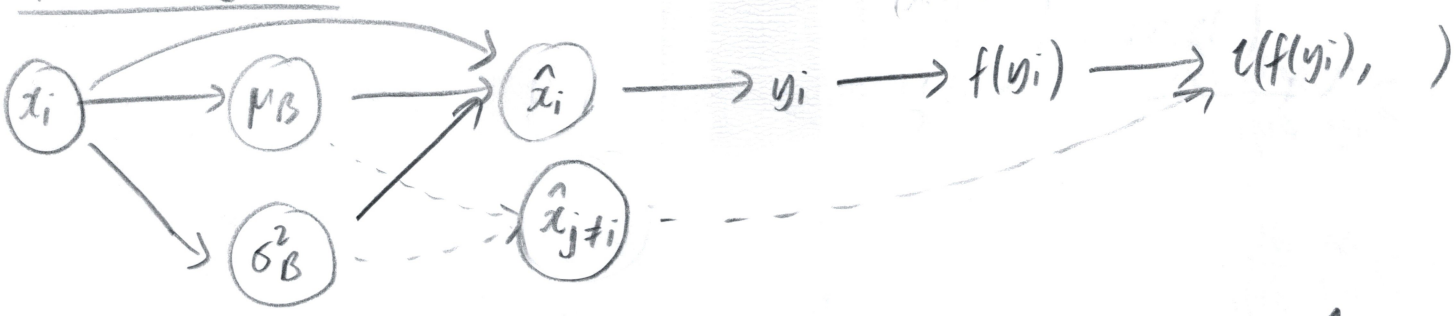
(*) Just because we consider one unit/activation $\not\equiv$ scalar fn.

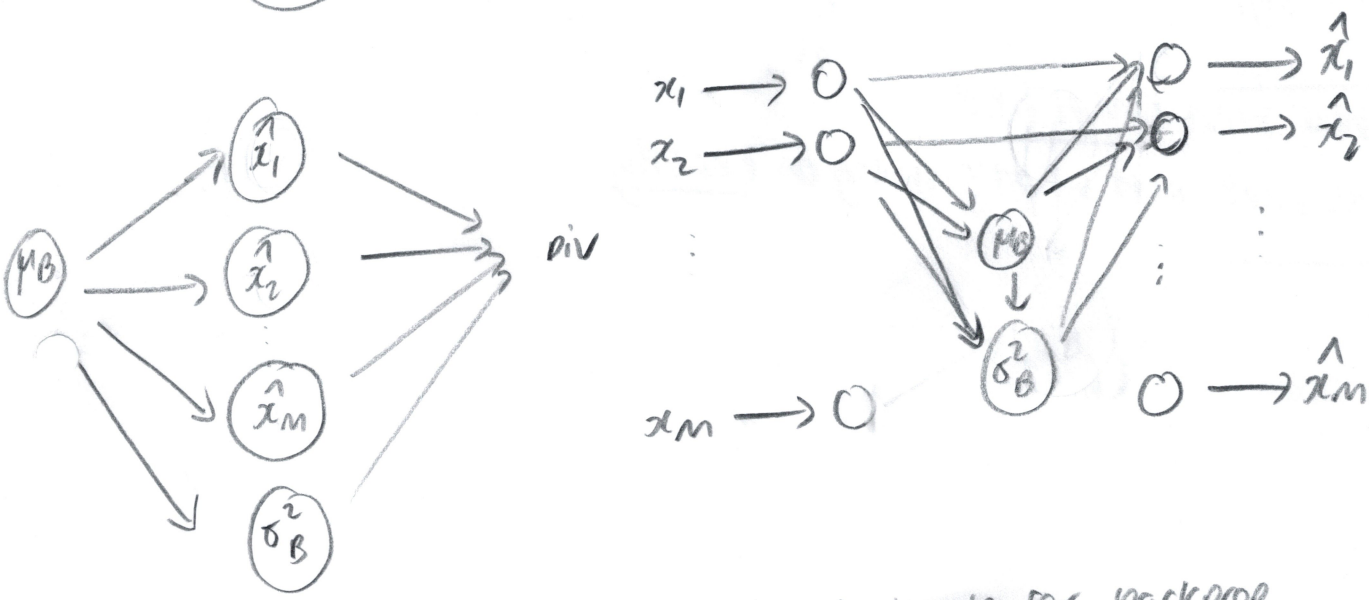
(*) $BN_{\gamma,\beta}(x_i) = f(x_i, x_{-i}, \mu_{\mathcal{B}}(x_{1:m}), \sigma^2_{\mathcal{B}}(x_{1:m}), \underbrace{\gamma,\beta}_{\text{indirect (?)}})$

(*) Vectors $\longrightarrow$ batch norm vector fm our minibatch.

Backpropagation (BN) - get adept at not having conf. by art. 21/05/2020

Influence diagram





(W): I don't fully understand the form of the chain rule for backprop in BN

(W): I can't fully visualise the influence chain, in particular why the lectures' divergence omits the late activations from the influence diagram.

- (W) How does this affect backprop algo with respect to

$$\nabla_{W_R} CE(\hat{y}, t) \text{ and } \nabla_{y_R} CE(\hat{y}, t)$$

(*) write down anyway (Ioffe & Szegedy) (need to reconcile with lec

$$\frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \frac{\partial y_i}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma$$

$$\frac{\partial \ell}{\partial \sigma^2_B} = \left(\sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma^2_B + \epsilon}} \right) + \frac{\partial \ell}{\partial \sigma^2_B} \cdot \frac{\sum_{i=1}^{m} -2(x_i - \mu_B)}{m}$$

$$\frac{\partial l}{\partial \mu_B} = \frac{\partial l}{\partial x_i} = \frac{\partial l}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\sigma_B^2 + \epsilon}} + \frac{\partial l}{\partial \sigma_B^2} \cdot \frac{2(x_i - \mu_B)}{m} + \frac{\partial l}{\partial \mu_B} \cdot \frac{1}{m}$$

$$\frac{\partial l}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i} \cdot \hat{x}_i$$

$$\frac{\partial l}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i}$$

(*) These have been simplified

(*) Need to work out what's going on exactly
