UNIVERSITY OF CAMBRIDGE

Dropout in RNNs Following a VI Interpretation

Yarin Gal

yg279@cam.ac.uk

Recurrent neural networks (RNNs) are damn useful.

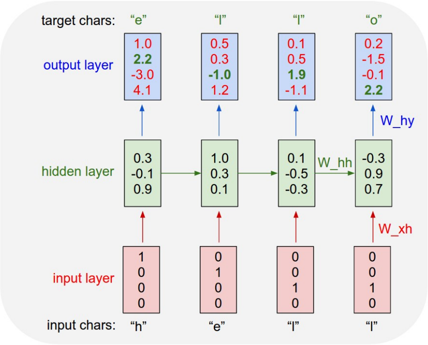


Figure : RNN structure

Image Source: karpathy.github.io/2015/05/21/rnn-effectiveness

But these also overfit very quickly...

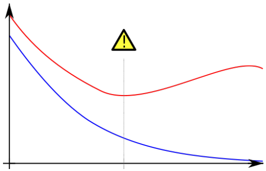


Figure : Overfitting

This means...

- We can't use large models
- We have to use early stopping
- We can't use small data
- We have to waste data for validation sets...

But these also overfit very quickly...

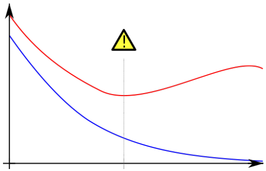


Figure : Overfitting

This means...

- We can't use large models
- We have to use early stopping
- We can't use small data
- We have to waste data for validation sets...

But these also overfit very quickly...

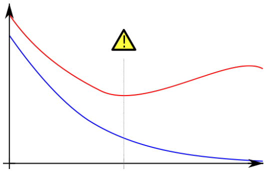


Figure : Overfitting

This means...

- We can't use large models
- We have to use early stopping
- We can't use small data
- We have to waste data for validation sets...

But these also overfit very quickly...

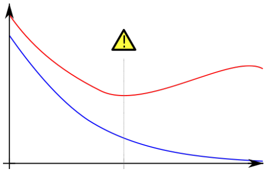


Figure : Overfitting

This means...

- We can't use large models
- We have to use early stopping
- We can't use small data
- We have to waste data for validation sets...

But these also overfit very quickly...

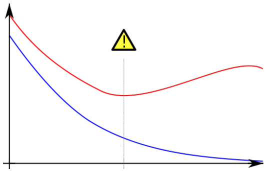


Figure : Overfitting

This means...

- We can't use large models
- We have to use early stopping
- We can't use small data
- We have to waste data for validation sets...

Dropout in recurrent neural networks

Let's use dropout then. But lots of research has claimed that that's a **bad idea**:

- **Pachitariu & Sahani**, 2013
  - noise added in the recurrent connections of an RNN leads to model instabilities
- **Bayer et al.**, 2013
  - with dropout, the RNNs dynamics change dramatically
- **Pham et al.**, 2014
  - dropout in recurrent layers disrupts the RNNs ability to model sequences
- **Zaremba et al.**, 2014
  - applying dropout to the non-recurrent connections alone results in improved performance
- **Bluche et al.**, 2015
  - exploratory analysis of the performance of dropout before, inside, and after the RNNs

$\rightarrow$ has settled on using dropout for **inputs and outputs alone**:

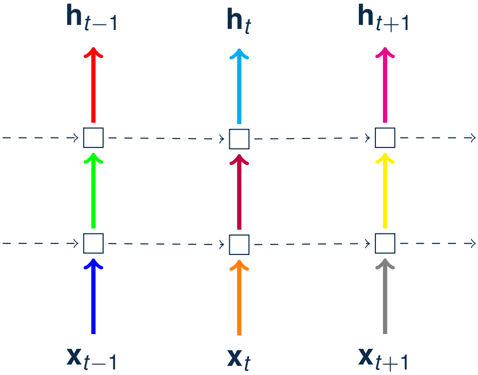


Figure : Naive application of dropout in RNNs (colours = different dropout masks)

Dropout in recurrent neural networks

Why not use dropout with recurrent layers?

- ~~It doesn't work~~
- ~~Noise drowns the signal~~
- Because it's not used correctly?

Dropout in recurrent neural networks

UNIVERSITY OF CAMBRIDGE

Why not use dropout with recurrent layers?

- ~~It doesn't work~~
- ~~Noise drowns the signal~~
- Because it's not used correctly?

Dropout in recurrent neural networks

Why not use dropout with recurrent layers?

- ~~It doesn't work~~
- ~~Noise drowns the signal~~
- Because it's not used correctly?

Dropout in recurrent neural networks

UNIVERSITY OF CAMBRIDGE

Why not use dropout with recurrent layers?

- ~~It doesn't work~~
- ~~Noise drowns the signal~~
- Because it's not used correctly?

First, some background on Bayesian modelling and VI in Bayesian neural networks.

Bayesian modelling and inference

UNIVERSITY OF CAMBRIDGE

- Observed inputs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$
- Capture stochastic process believed to have generated outputs
- Def. ω model parameters as r.v.
- Prior dist. over ω: $p(\omega)$
- Likelihood: $p(\mathbf{Y}|\omega, \mathbf{X})$
- Posterior: $p(\omega|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\omega,\mathbf{X})p(\omega)}{p(\mathbf{Y}|\mathbf{X})}$ (Bayes' theorem)
- Predictive distribution given new input $\mathbf{x}^*$

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \omega) \underbrace{\boxed{p(\omega|\mathbf{X}, \mathbf{Y})}}_{\text{posterior}} \mathrm{d}\omega$$

- But... $p(\omega|\mathbf{X}, \mathbf{Y})$ is often intractable

Bayesian modelling and inference

UNIVERSITY OF CAMBRIDGE

- Observed inputs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$
- Capture stochastic process believed to have generated outputs
- Def. ω model parameters as r.v.
- Prior dist. over ω: $p(\omega)$
- Likelihood: $p(\mathbf{Y}|\omega, \mathbf{X})$
- Posterior: $p(\omega|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\omega,\mathbf{X})p(\omega)}{p(\mathbf{Y}|\mathbf{X})}$ (Bayes' theorem)
- Predictive distribution given new input $\mathbf{x}^*$

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \omega) \underbrace{\boxed{p(\omega|\mathbf{X}, \mathbf{Y})}}_{\text{posterior}} \mathrm{d}\omega$$

- But... $p(\omega|\mathbf{X}, \mathbf{Y})$ is often intractable

UNIVERSITY OF CAMBRIDGE

- Observed inputs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$
- Capture stochastic process believed to have generated outputs
- Def. ω model parameters as r.v.
- Prior dist. over ω: $p(\omega)$
- Likelihood: $p(\mathbf{Y}|\omega, \mathbf{X})$
- Posterior: $p(\omega|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\omega,\mathbf{X})p(\omega)}{p(\mathbf{Y}|\mathbf{X})}$ (Bayes' theorem)
- Predictive distribution given new input $\mathbf{x}^*$

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \omega) \underbrace{\boxed{p(\omega|\mathbf{X}, \mathbf{Y})}}_{\text{posterior}} \mathrm{d}\omega$$

- But... $p(\omega|\mathbf{X}, \mathbf{Y})$ is often intractable

UNIVERSITY OF CAMBRIDGE

- Approximate $p(\omega|\mathbf{X},\mathbf{Y})$ with simple dist. $q_\theta(\omega)$
- Minimise divergence from posterior w.r.t. θ

$$\mathrm{KL}(q_\theta(\omega)\ ||\ p(\omega|\mathbf{X},\mathbf{Y}))$$

- Identical to minimising

$$\mathcal{L}_{\mathrm{VI}}(\theta) := -\int q_\theta(\omega)\log \overbrace{p(\mathbf{Y}|\mathbf{X},\omega)}^{\text{likelihood}} d\omega + \mathrm{KL}(q_\theta(\omega)||\overbrace{p(\omega)}^{\text{prior}})$$

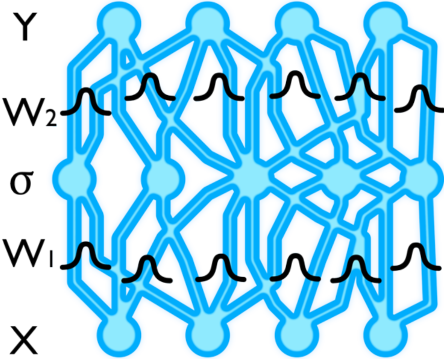
- We can approximate the **predictive distribution**

$$q_\theta(\mathbf{y}^*|\mathbf{x}^*) = \int p(\mathbf{y}^*|\mathbf{x}^*,\omega)q_\theta(\omega)d\omega.$$

- Place prior $p(\mathbf{W}_i)$:

$$\mathbf{W}_i \sim \mathcal{N}(0, \mathbf{I})$$

for $i \leq L$ (and write $\omega := \{\mathbf{W}_i\}_{i=1}^{L}$).



- Output is a r.v. $\mathbf{f}(\mathbf{x}, \omega) = \mathbf{W}_L \sigma(\ldots \mathbf{W}_2 \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \ldots)$.

Bayesian neural networks

- Place prior $p(\mathbf{W}_i)$:

$$\mathbf{W}_i \sim \mathcal{N}(0, \mathbf{I})$$

  for $i \leq L$ (and write $\omega := \{\mathbf{W}_i\}_{i=1}^{L}$).

- Output is a r.v. $\mathbf{f}\big(\mathbf{x}, \omega\big) = \mathbf{W}_L \sigma\big(...\mathbf{W}_2 \sigma\big(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1\big)...\big)$.

- Softmax likelihood for class.: $p\big(y|\mathbf{x}, \omega\big) = \text{softmax}\left(\mathbf{f}\big(\mathbf{x}, \omega\big)\right)$
  or a Gaussian for regression: $p\big(\mathbf{y}|\mathbf{x}, \omega\big) = \mathcal{N}\left(\mathbf{y}; \mathbf{f}\big(\mathbf{x}, \omega\big), \tau^{-1}\mathbf{I}\right)$.

- But difficult to evaluate posterior

$$p\big(\omega|\mathbf{X}, \mathbf{Y}\big).$$

UNIVERSITY OF CAMBRIDGE

- Place prior $p(\mathbf{W}_i)$:

$$\mathbf{W}_i \sim \mathcal{N}(0, \mathbf{I})$$

  for $i \leq L$ (and write $\omega := \{\mathbf{W}_i\}_{i=1}^{L}$).

- Output is a r.v. $\mathbf{f}\big(\mathbf{x}, \omega\big) = \mathbf{W}_L \sigma\big(...\mathbf{W}_2 \sigma\big(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1\big)...\big)$.

- Softmax likelihood for class.: $p\big(y|\mathbf{x}, \omega\big) = \text{softmax}\left(\mathbf{f}\big(\mathbf{x}, \omega\big)\right)$
  or a Gaussian for regression: $p\big(\mathbf{y}|\mathbf{x}, \omega\big) = \mathcal{N}\left(\mathbf{y}; \mathbf{f}\big(\mathbf{x}, \omega\big), \tau^{-1}\mathbf{I}\right)$.

- But difficult to evaluate posterior

$$p\big(\omega|\mathbf{X}, \mathbf{Y}\big).$$

- Place prior $p(\mathbf{W}_i)$:
  $$\mathbf{W}_i \sim \mathcal{N}(0, \mathbf{I})$$
  for $i \leq L$ (and write $\omega := \{\mathbf{W}_i\}_{i=1}^{L}$).
- Output is a r.v. $\mathbf{f}\big(\mathbf{x}, \omega\big) = \mathbf{W}_L \sigma\big(...\mathbf{W}_2 \sigma\big(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1\big)...\big)$.
- Softmax likelihood for class.: $p\big(y|\mathbf{x}, \omega\big) = \text{softmax}\left(\mathbf{f}\big(\mathbf{x}, \omega\big)\right)$
  or a Gaussian for regression: $p\big(\mathbf{y}|\mathbf{x}, \omega\big) = \mathcal{N}\left(\mathbf{y}; \mathbf{f}\big(\mathbf{x}, \omega\big), \tau^{-1}\mathbf{I}\right)$.
- But difficult to evaluate posterior
  $$p\big(\omega|\mathbf{X}, \mathbf{Y}\big).$$

UNIVERSITY OF CAMBRIDGE

- Def $q_\theta(\omega)$ to approximate posterior $p(\omega|\mathbf{X}, \mathbf{Y})$
- KL divergence to minimise:

$$\begin{aligned} &\mathrm{KL}\big(q_\theta(\omega) \,||\, p(\omega|\mathbf{X}, \mathbf{Y})\big) \\ &\quad \propto \boxed{-\int q_\theta(\omega) \log p(\mathbf{Y}|\mathbf{X}, \omega) \mathrm{d}\omega} + \mathrm{KL}\big(q_\theta(\omega) \,||\, p(\omega)\big) \\ &\quad =: \mathcal{L}(\theta) \end{aligned}$$

- Approximate the integral with MC integration $\widehat{\omega} \sim q_\theta(\omega)$:

$$\widehat{\mathcal{L}}(\theta) := -\log p(\mathbf{Y}|\mathbf{X}, \widehat{\omega}) + \mathrm{KL}\big(q_\theta(\omega) \,||\, p(\omega)\big)$$

- Def $q_\theta(\omega)$ to approximate posterior $p(\omega|\mathbf{X},\mathbf{Y})$
- KL divergence to minimise:

$$
\begin{aligned}
&\mathrm{KL}\big(q_\theta(\omega)\;||\;p(\omega|\mathbf{X},\mathbf{Y})\big)\\
&\quad\propto \boxed{-\int q_\theta(\omega)\log p(\mathbf{Y}|\mathbf{X},\omega)\mathrm{d}\omega} + \mathrm{KL}\big(q_\theta(\omega)\;||\;p(\omega)\big)\\
&\quad=: \mathcal{L}(\theta)
\end{aligned}
$$

- Approximate the integral with MC integration $\widehat{\omega} \sim q_\theta(\omega)$:

$$\widehat{\mathcal{L}}(\theta) := -\log p(\mathbf{Y}|\mathbf{X},\widehat{\omega}) + \mathrm{KL}\big(q_\theta(\omega)\;||\;p(\omega)\big)$$

- Def $q_\theta(\omega)$ to approximate posterior $p(\omega|\mathbf{X},\mathbf{Y})$
- KL divergence to minimise:

$$
\begin{aligned}
&\mathrm{KL}\big(q_\theta(\omega) \;||\; p(\omega|\mathbf{X},\mathbf{Y})\big)\\
&\quad \propto \boxed{-\int q_\theta(\omega)\log p(\mathbf{Y}|\mathbf{X},\omega)\mathrm{d}\omega} + \mathrm{KL}\big(q_\theta(\omega) \;||\; p(\omega)\big)\\
&\quad =: \mathcal{L}(\theta)
\end{aligned}
$$

- Approximate the integral with MC integration $\widehat{\omega} \sim q_\theta(\omega)$:

$$
\widehat{\mathcal{L}}(\theta) := -\log p(\mathbf{Y}|\mathbf{X},\widehat{\omega}) + \mathrm{KL}\big(q_\theta(\omega) \;||\; p(\omega)\big)
$$

- Unbiased estimator:

$$E_{\widehat{\omega} \sim q_\theta(\omega)}\left(\widehat{\mathcal{L}}(\theta)\right) = \mathcal{L}(\theta)$$

- Converges to the same optima as $\mathcal{L}(\theta)$
- For inference, repeat:
  - Sample $\widehat{\omega} \sim q_\theta(\omega)$
  - And minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p(\mathbf{Y}|\mathbf{X}, \widehat{\omega}) + \mathrm{KL}(q_\theta(\omega) \mid\mid p(\omega))$$

  w.r.t. θ.

- Unbiased estimator:

$$E_{\widehat{\omega}\sim q_\theta(\omega)}\big(\widehat{\mathcal{L}}(\theta)\big) = \mathcal{L}(\theta)$$

- Converges to the same optima as $\mathcal{L}(\theta)$

- For inference, repeat:
  - Sample $\widehat{\omega} \sim q_\theta(\omega)$
  - And minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p(\mathbf{Y}|\mathbf{X}, \widehat{\omega}) + \mathrm{KL}\big(q_\theta(\omega)\;||\;p(\omega)\big)$$

w.r.t. θ.

- Unbiased estimator:

$$E_{\widehat{\omega} \sim q_\theta(\omega)}\big(\widehat{\mathcal{L}}(\theta)\big) = \mathcal{L}(\theta)$$

- Converges to the same optima as $\mathcal{L}(\theta)$
- For inference, repeat:
  - Sample $\widehat{\omega} \sim q_\theta(\omega)$
  - And minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p\big(\mathbf{Y}|\mathbf{X}, \widehat{\omega}\big) + \mathrm{KL}\big(q_\theta(\omega) \mid\mid p(\omega)\big)$$

  w.r.t. θ.

UNIVERSITY OF CAMBRIDGE

- Given variational parameters $\theta = \left\{[\mathbf{m}_{i1}, ..., \mathbf{m}_{iK}]\right\}_{i=1}^{L}$:

$$q_\theta(\omega) = \prod_i q_\theta(\mathbf{W}_i)$$

$$q_\theta(\mathbf{W}_i) = \prod_k q_{\mathbf{m}_{ik}}(\mathbf{w}_{ik})$$

$$q_{\mathbf{m}_{ik}}(\mathbf{w}_{ik}) = p\mathcal{N}(0, \sigma^2) + (1-p)\mathcal{N}(\mathbf{m}_{ik}, \sigma^2)$$

$\rightarrow$ k'th column of the i'th layer is a multivariate mixture of Gaussians

- With small enough σ^2, in practice equivalent to

$$\mathbf{z}_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1}$$
$$\mathbf{W}_i = \mathbf{M}_i \cdot \text{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i})$$

with $\mathbf{z}_{i,j}$ Bernoulli r.v.s.

In summary:

Minimise divergence between $q_\theta(\omega)$ and $p(\omega|\mathbf{X}, \mathbf{Y})$:

- Repeat:
  - Sample $\widehat{\mathbf{z}}_{i,j} \sim \text{Bernoulli}(p_i)$ and set

$$\widehat{\mathbf{W}}_i = \mathbf{M}_i \cdot \text{diag}([\widehat{\mathbf{z}}_{i,j}]_{j=1}^{K_i})$$
$$\widehat{\omega} = \{\widehat{\mathbf{W}}_i\}_{i=1}^{L}$$

  - Minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p\big(\mathbf{Y}|\mathbf{X}, \widehat{\omega}\big) + \text{KL}\big(q_\theta(\omega) \,||\, p(\omega)\big)$$

w.r.t. $\theta = \{\mathbf{M}_i\}_{i=1}^{L}$ (set of matrices).

In summary:

Minimise divergence between $q_\theta(\omega)$ and $p(\omega|\mathbf{X}, \mathbf{Y})$:

- Repeat:
  - = Randomly set columns of $\mathbf{M}_i$ to zero
  - Minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p\big(\mathbf{Y}|\mathbf{X}, \widehat{\omega}\big) + \mathrm{KL}\big(q_\theta(\omega) \,||\, p(\omega)\big)$$

w.r.t. $\theta = \{\mathbf{M}_i\}_{i=1}^L$ (set of matrices).

In summary:

Minimise divergence between $q_\theta(\omega)$ and $p(\omega|\mathbf{X}, \mathbf{Y})$:

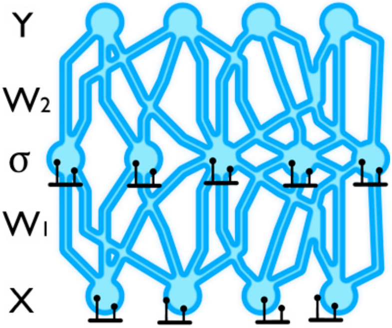
- Repeat:
  - = Randomly set units of the network to zero
  - Minimise (one step)

$$\widehat{\mathcal{L}}(\theta) = -\log p\big(\mathbf{Y}|\mathbf{X}, \widehat{\omega}\big) + \mathrm{KL}\big(q_\theta(\omega) \;||\; p(\omega)\big)$$

w.r.t. $\theta = \{\mathbf{M}_i\}_{i=1}^{L}$ (set of matrices).

UNIVERSITY OF CAMBRIDGE

Sounds familiar?



$$\widehat{\mathcal{L}}(\theta) = \overbrace{-\log p\big(\mathbf{Y}|\mathbf{X}, \widehat{\omega}\big)}^{=\text{ loss}} + \overbrace{\mathrm{KL}\big(q_\theta(\omega)\ ||\ p(\omega)\big)}^{=\ L_2\text{ reg}}$$

Implementing VI with $q_\theta(\cdot)$ above = implementing dropout in deep network

- Multiplicative Gaussian noise (Srivastava et al. 2014) –
- Multiply network units by $\mathcal{N}(1, 1)$
- Same performance as dropout

$$\Updownarrow$$

Multiplicative Gaussian noise as approximate inference[1]

$$\mathbf{z}_{i,j} \sim \mathcal{N}(1, 1) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1}$$
$$\mathbf{W}_i = \mathbf{M}_i \cdot \mathrm{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i})$$
$$q_\theta(\omega) = \prod q_{\mathbf{M}_i}(\mathbf{W}_i)$$

Similarly for **drop-connect** (Wan et al., 2013), etc.

[1] See Gal and Ghahramani (2015) and Kingma et al. (2015)

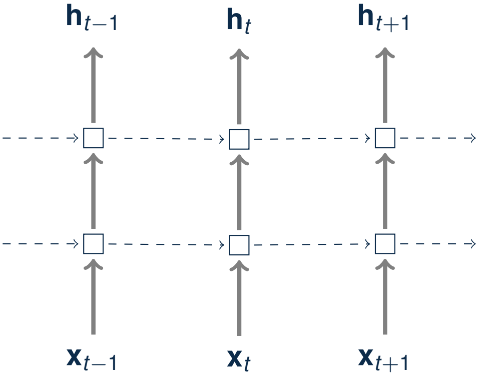


Figure : A Recurrent Neural Network

Now, in RNNs...

- Input sequence of vectors $\mathbf{x} = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$ with T time steps
- Let ω = $\{$**all** weight matrices in the model$\}$
- Define $\mathbf{h}_t = \mathbf{f}^{\omega}_{\mathbf{h}}(\mathbf{x}_t, \mathbf{h}_{t-1})$
  - **single** recurrent unit transition. E.g. tanh of affine transformation: $\tanh(W\mathbf{x}_t + U\mathbf{h}_{t-1} + b)$
- Set $\mathbf{f}^{\omega}_{\mathbf{y}}(\mathbf{h}_T) = \mathbf{f}^{\omega}_{\mathbf{y}}(\mathbf{f}^{\omega}_{\mathbf{h}}(\mathbf{x}_T, ...\mathbf{f}^{\omega}_{\mathbf{h}}(\mathbf{x}_1, \mathbf{h}_0)...))$
  - model output (e.g. affine transformation of last state, or function of all states)
- Lastly, define $p(\mathbf{y}|\mathbf{f}^{\omega}_{\mathbf{y}}(\mathbf{h}_T))$
  - model likelihood. E.g. $\mathcal{N}(\mathbf{y}; \mathbf{f}^{\omega}_{\mathbf{y}}(\mathbf{h}_T), \sigma^2)$
- Similarly for LSTM, GRU

UNIVERSITY OF CAMBRIDGE

- Looking at the variational lower bound, we have:

$$\int q(\omega) \log p(\mathbf{y}|\mathbf{f}^{\omega}_{\mathbf{y}}(\mathbf{h}_T))\mathrm{d}\omega =$$

$$\int q(\omega) \log p\left(\mathbf{y}\middle|\mathbf{f}^{\omega}_{\mathbf{y}}\big(\mathbf{f}^{\omega}_{\mathbf{h}}(\mathbf{x}_T, ...\mathbf{f}^{\omega}_{\mathbf{h}}(\mathbf{x}_1, \mathbf{h}_0)...)\big)\right)\mathrm{d}\omega,$$

- Using MC integration with $\widehat{\omega} \sim q(\omega)$,

$$\mathcal{L}_{VI} \approx -\log p\left(\mathbf{y}\middle|\mathbf{f}^{\widehat{\omega}}_{\mathbf{y}}\big(\mathbf{f}^{\widehat{\omega}}_{\mathbf{h}}(\mathbf{x}_T, ...\mathbf{f}^{\widehat{\omega}}_{\mathbf{h}}(\mathbf{x}_1, \mathbf{h}_0)...)\big)\right)$$

$$+ \mathrm{KL}\big(q_\theta(\omega) \mid\mid p(\omega)\big).$$

Dropout in RNNs

Objective:

$$-\log p\left(\mathbf{y}\,\middle|\,\mathbf{f}_{\mathbf{y}}^{\boxed{\widehat{\omega}}}(\mathbf{f}_{\mathbf{h}}^{\boxed{\widehat{\omega}}}(\mathbf{x}_T, \ldots \mathbf{f}_{\mathbf{h}}^{\boxed{\widehat{\omega}}}(\mathbf{x}_1, \mathbf{h}_0)\ldots))\right) + \ldots \quad \boxed{\widehat{\omega}} \sim q(\omega)$$

- In practice, **use the same dropout mask at each time step**

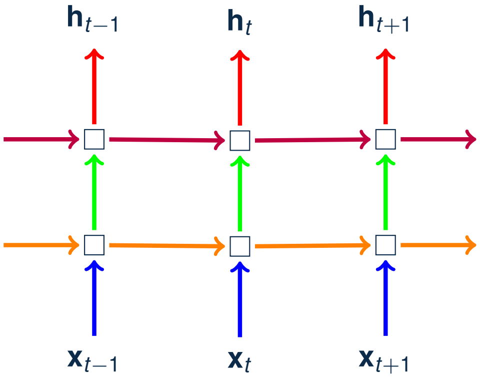


Figure : Bayesian motivated dropout in RNNs (colours = dropout masks)

Word embedding dropout

UNIVERSITY OF CAMBRIDGE

- With **continuous** inputs we apply dropout to the input layer (place a distr. over weight matrix)
- But not for models with **discrete** inputs...
- Word embeddings: input can be seen as either the word embed. itself, or a "**one-hot**" encoding times an embed. matrix
- Optimising embedding matrix can lead to **overfitting**...
- Let's apply dropout **to the one-hot encoded vectors**

Word embedding dropout

UNIVERSITY OF CAMBRIDGE

- **In practice**, drop words at random throughout the sentence
  - Randomly set embedding matrix **rows** to zero – entire word embeddings
  - Mask is **repeated** at each time step $\rightarrow$ **drop the same words throughout the sequence**
  - i.e. drop word **types** at random rather than word **tokens**
- For **example**, “the dog and the cat” might become “— dog and — cat” or “the — and the cat”, but never “— dog and the cat”.
- Can be interpreted as encouraging model to **not “depend” on single words.**

Some results (much more in paper):

- Sentiment analysis (Pang & Lee, 2005)

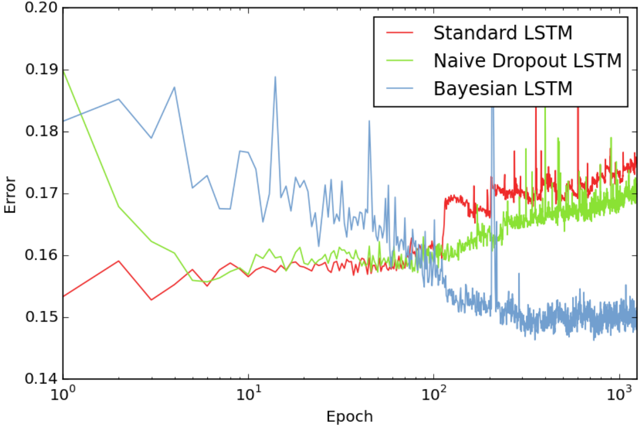


Figure : LSTM test error

Working dropout in recurrent layers

Some results (much more in paper):

- Sentiment analysis (Pang & Lee, 2005)

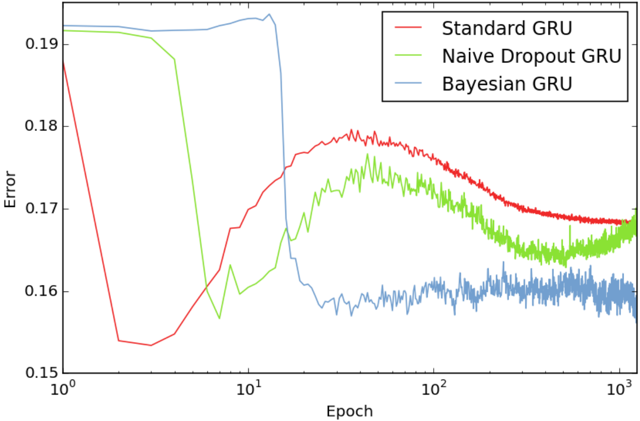


Figure : GRU test error

- Language model (Penn Treebank)

Some results (much more in paper):

- Sentiment analysis (Pang & Lee, 2005)
- Language model (Penn Treebank)

	Medium LSTM			Large LSTM		
	Validation	Test	WPS	Validation	Test	WPS
Non-regularized (early stopping)	121.1	121.7	5.5K	128.3	127.4	2.5K
Moon et al. [19]	100.7	97.0	4.8K	122.9	118.7	3K
Moon et al. [19] +emb dropout	88.9	86.5	4.8K	88.8	86.0	3K
Zaremba et al. [4]	86.2	82.7	5.5K	82.2	78.4	2.5K
Variational (tied weights)	81.8 ± 0.2	79.7 ± 0.1	4.7K	77.3 ± 0.2	75.0 ± 0.1	2.4K
Variational (tied weights, MC)	–	79.0 ± 0.1	–	–	74.1 ± 0.0	–
Variational (untied weights)	81.9 ± 0.2	79.7 ± 0.1	2.7K	77.9 ± 0.3	75.2 ± 0.2	1.6K
Variational (untied weights, MC)	–	$\mathbf{78.6 \pm 0.1}$	–	–	$\mathbf{73.4 \pm 0.0}$	–

Working dropout in recurrent layers

Some results (much more in paper):

- Sentiment analysis (Pang & Lee, 2005)
- Language model (Penn Treebank)

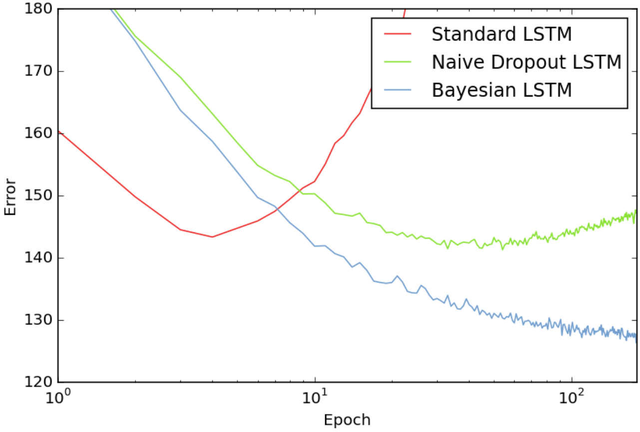


Figure : 2 layers LSTM, 200 units

Many unanswered questions left

- Practical deep learning uncertainty?
  - Capture language ambiguity?

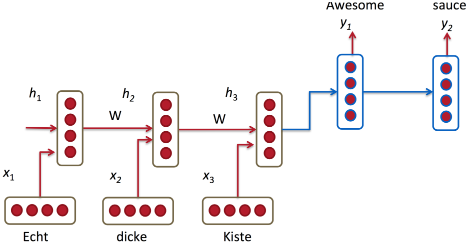
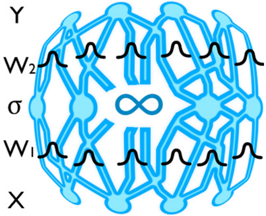


Image Source: `cs224d.stanford.edu/lectures/CS224d-Lecture8.pdf`

  - Weight uncertainty for model debugging?
- Principled extensions of deep learning?
  - New appr. distributions = new stochastic reg. techniques?
  - Model compression: $W_i \sim$ discrete distribution w. continuous base measure?

Many unanswered questions left

- Practical deep learning uncertainty?
  - Capture language ambiguity?
  - Weight uncertainty for model debugging?



- Principled extensions of deep learning?
  - New appr. distributions = new stochastic reg. techniques?
  - Model compression: $W_i \sim$ discrete distribution w. continuous base measure?

Many unanswered questions left

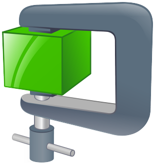
- Practical deep learning uncertainty?
  - Capture language ambiguity?
  - Weight uncertainty for model debugging?
- Principled extensions of deep learning?
  - New appr. distributions = new stochastic reg. techniques?

$$q_\theta(\omega) = ?$$

  - Model compression: $\mathbf{W}_i \sim$ discrete distribution w. continuous base measure?

Many unanswered questions left

- Practical deep learning uncertainty?
  - Capture language ambiguity?
  - Weight uncertainty for model debugging?
- Principled extensions of deep learning?
  - New appr. distributions = new stochastic reg. techniques?
  - Model compression: $\mathbf{W}_i \sim$ discrete distribution w. continuous base measure?



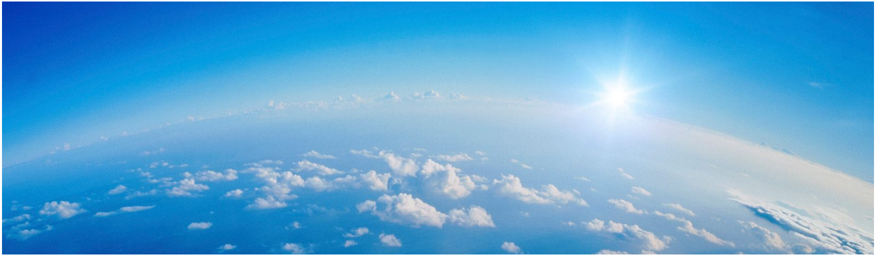
Many unanswered questions left

UNIVERSITY OF CAMBRIDGE

- Practical deep learning uncertainty?
  - Capture language ambiguity?
  - Weight uncertainty for model debugging?
- Principled extensions of deep learning?
  - New appr. distributions = new stochastic reg. techniques?
  - Model compression: $\mathbf{W}_i \sim$ discrete distribution w. continuous base measure?

Work in progress!

New horizons

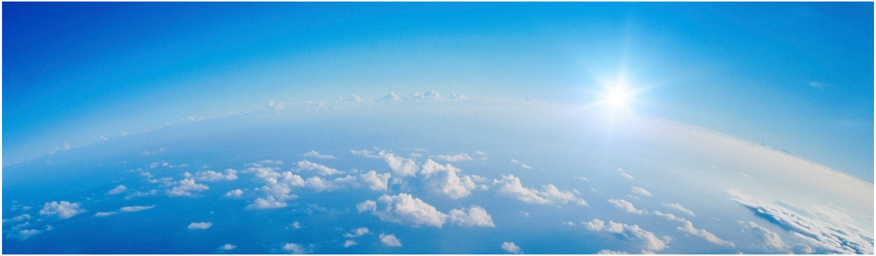


Most exciting is work to come:

- **Practical uncertainty** in deep learning applications
- **Principled extensions** to deep learning tools
- **Hybrid** deep learning – Bayesian models

and much, much, more.

New horizons



Most exciting is work to come:

- **Practical uncertainty** in deep learning applications
- **Principled extensions** to deep learning tools
- **Hybrid** deep learning – Bayesian models

and much, much, more.

Thank you for listening.