Conditional Modeling For Fun and Profit

Kyle Kastner

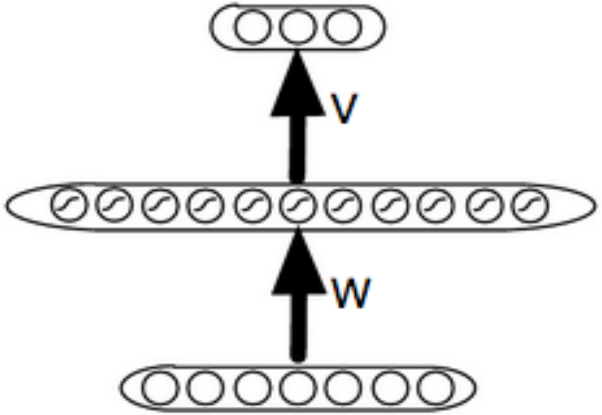
Université de Montréal - MILA
Intern - IBM Watson @ Yorktown Heights

Deep Learning, Simple Concepts

- Universal function approximators
- *Learn* the features
- Desire hierarchy in learned features
  - $y = h(g(f(x)))$
  - $\{h, g, f\}$ are nonlinear functions
- Classification
  - Learn $p(y \mid x) = h(g(f(x)))$

[1]

Basic Anatomy

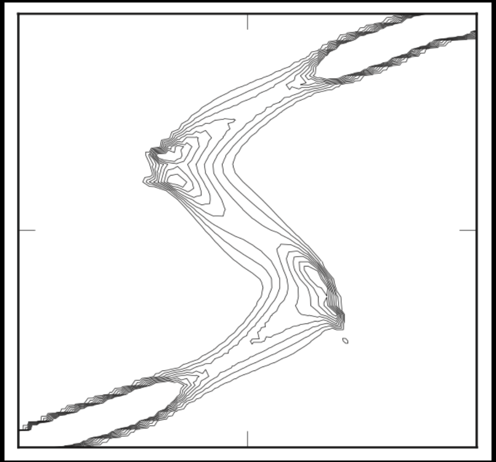


- Weights (**W, V**)
- Biases (**b, c**)
- Morph features using non-linear functions e.g.
  - layer_1_out = tanh(dot(X, W) + b)
  - layer_2_out = tanh(dot(layer_1_out, V) + c) ...
- Backpropagation to “step” values of **W**,**V**,**b**,**c**

[1, 2]

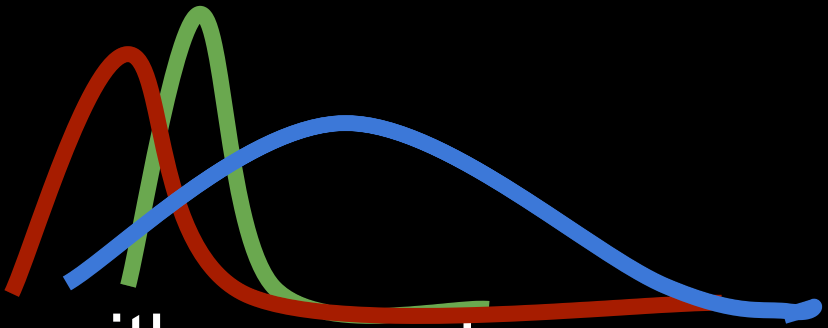
Mixture Density Networks

- What are sufficient statistics?
  - Describe an instance of a distribution
  - Gaussian with mean ***u***, variance ***s***
  - Bernoulli with probability ***p***
- Ties to neural networks
  - Arbitrary output parameters
  - Can we interpret parameters in a layer as sufficient statistics? YES!
  - Cost / regularization forces this relationship



[3, 1]

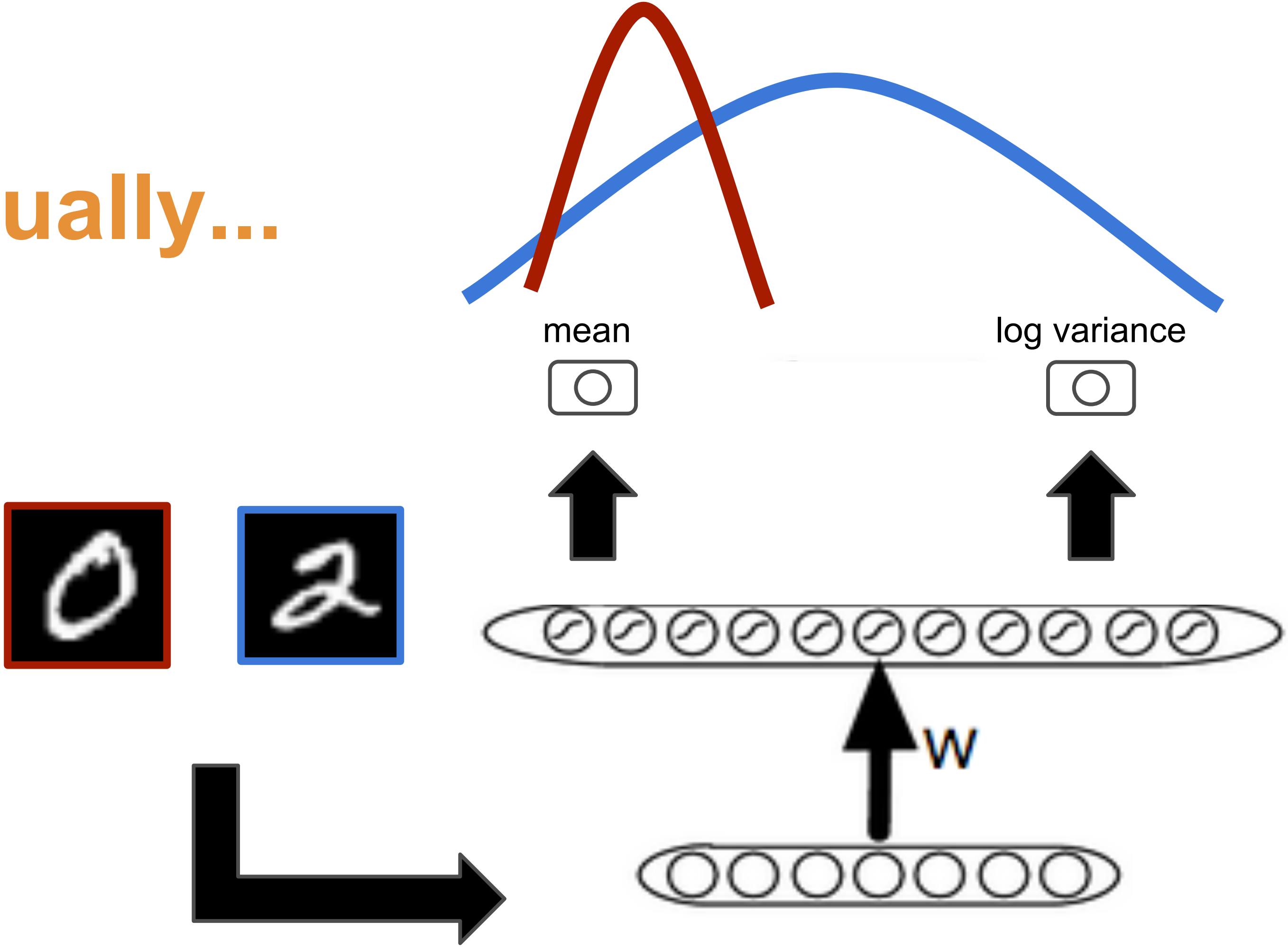
Parameterizing Distributions



- sigmoid -> Bernoulli
- softmax -> Multinomial
- linear, linear -> Gaussian with mean, log_var
- softmax, linear, linear -> Gaussian mixture
- Can combine with recurrence
  - Learned, dynamic distributions over sequences
  - *Incredibly* powerful

[3, 1, 4, 5, 6, 7, 8, 9]

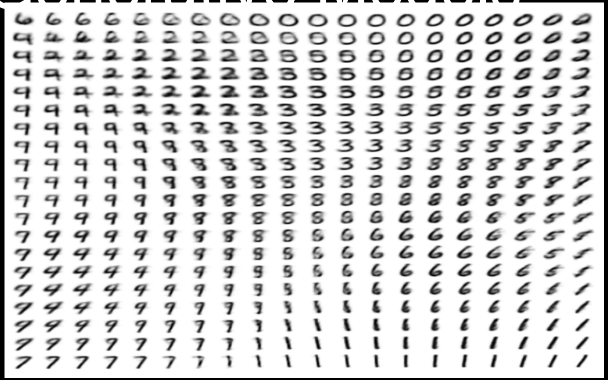
Visually...



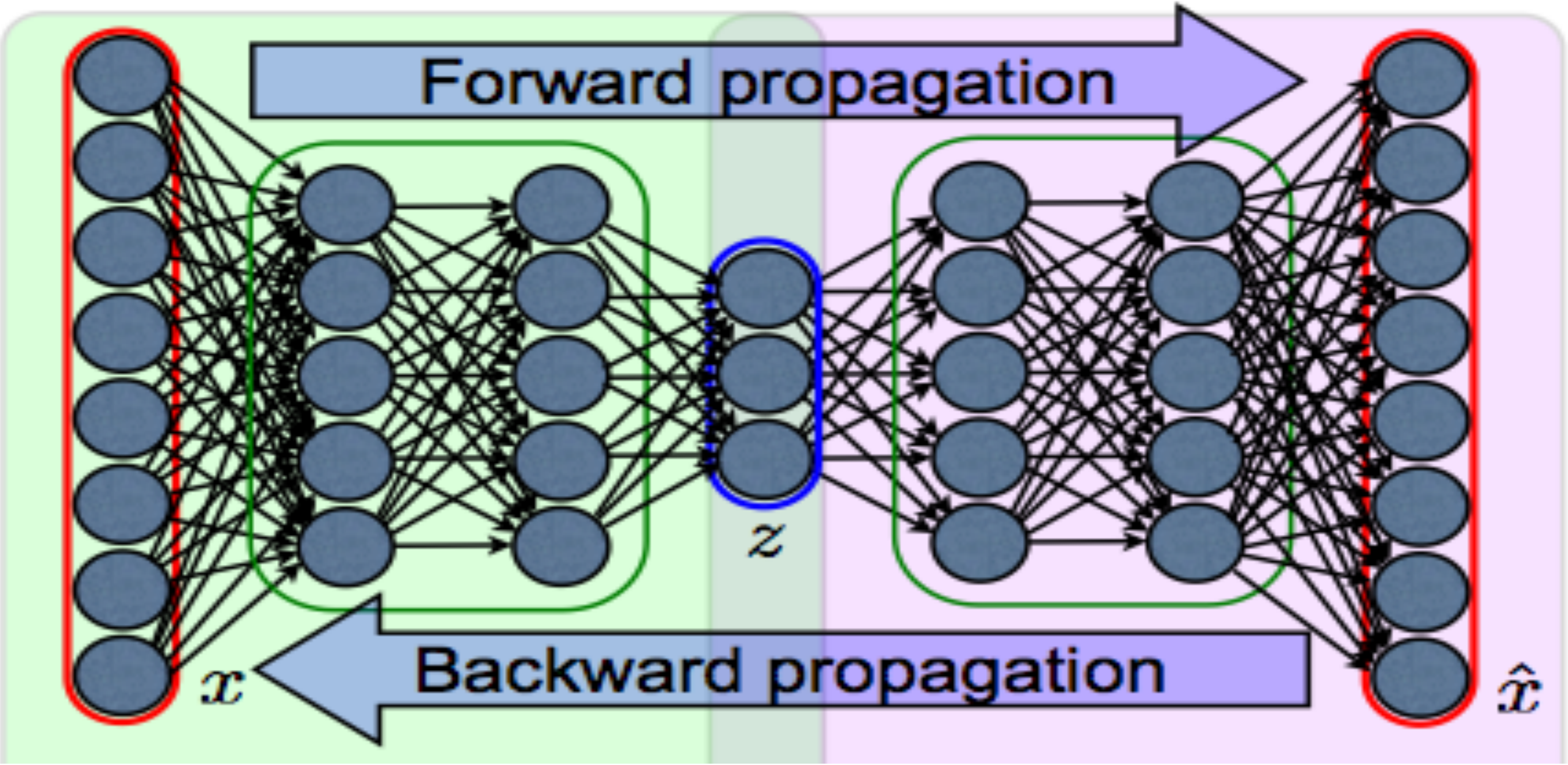
[1, 10]

Latent Factor Generative Models

- Auto-Encoding Variational Bayes
  D. Kingma and M. Welling
  - Model known as Variational Autoencoder (VAE)
  - See also Stochastic Backpropagation and Approximate Inference in Deep Generative Models
    Rezende, Mohamed, Wierstra

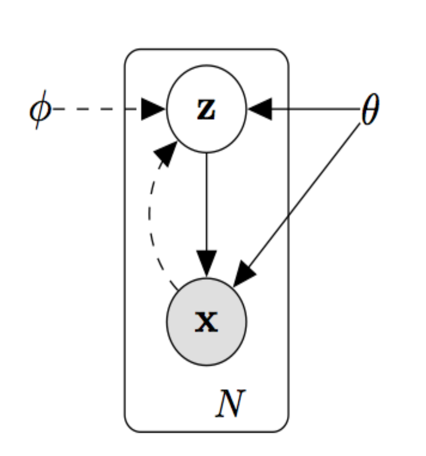


[11, 12, 13]



ENCODER [11, 12, 13] DECODER

A Bit About VAE

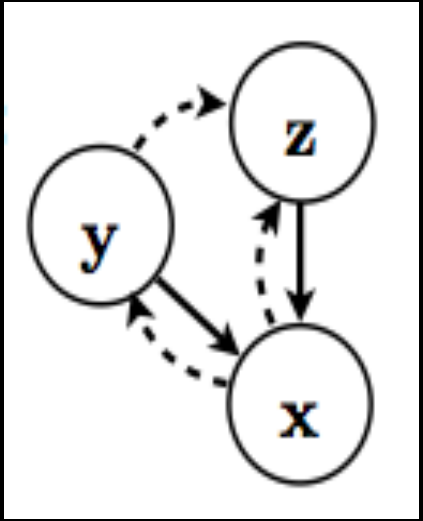


- Want to do latent variable modeling
- Don’t want to do MCMC or EM
- Sampling Z blocks gradient
- Reparameterization trick
  - Exact soln intractable for complex transforms (like NN)
  - Lower bound on likelihood with KL divergence
  - N(mu, sigma) -> mu + sigma * N(0, 1)
  - Like mixture density networks, but in the middle
  - Now trainable by backprop

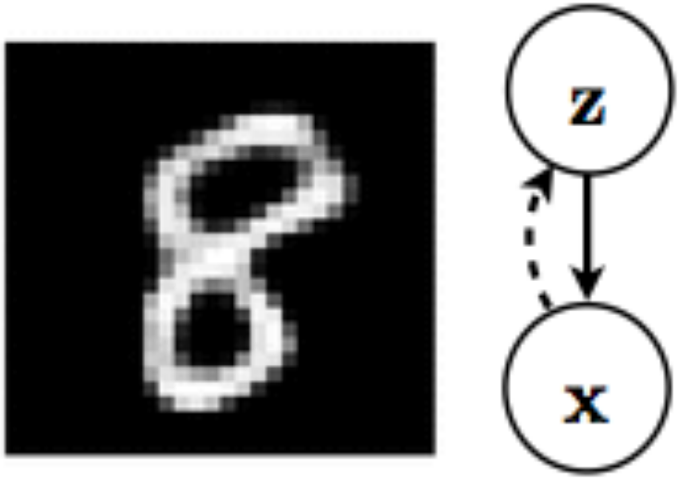
[11, 12, 13]

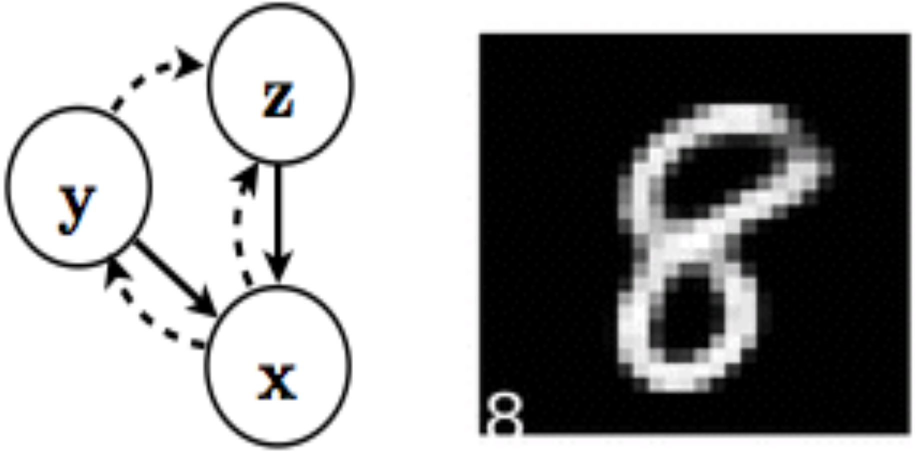
Taking The Wheel

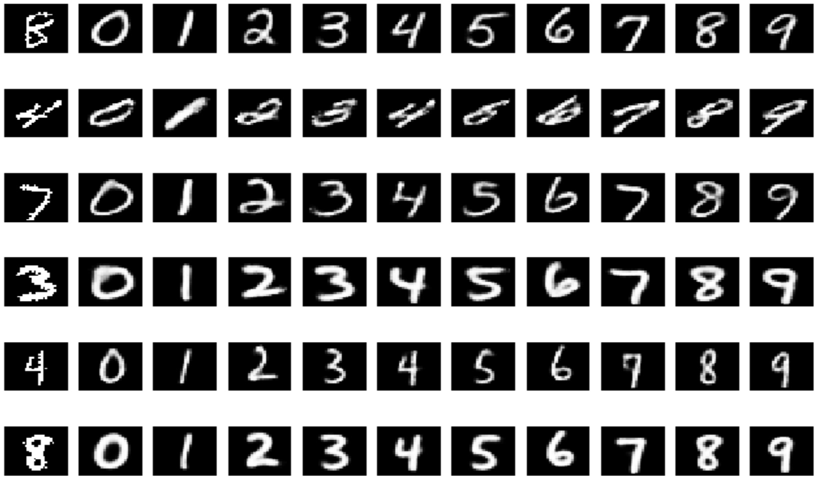
- Specifics of MNIST digits
  - Writing style and class
  - Traits are semi-independent
  - Can encode this in the model
  - y -> softmax classifier (~y is sample)
  - p(z | x, y), p(z | x, ~y) or p(z | x, f(x))
- Fully conditional version of M2
  - Semi-Supervised Learning with Deep Generative Models, Kingma, Rezende, Mohamed, Welling



[13, 14]







Conditioning, Visually

[13, 14]

In Practice...

- Conditioning is a *strong* signal
    - p(x_hat | z) vs. p(x_hat | z, y)
- Can give *control* or add prior knowledge
- Classification is an even stronger form
    - Prediction is learned by maximizing p(y | x) !
    - In classification, don’t worry about forming a useful z

[1, 13, 14]

Conditioning Feedforward

- Concatenate features
  - concatenate((X_train, conditioning), axis=1)
  - p(y | X_1 … X_n, L_1 … L_n)
- One hot label L (scikit-learn label_binarize)
- Could also be real valued
- Concat followed with multiple layers to “mix”

[1]

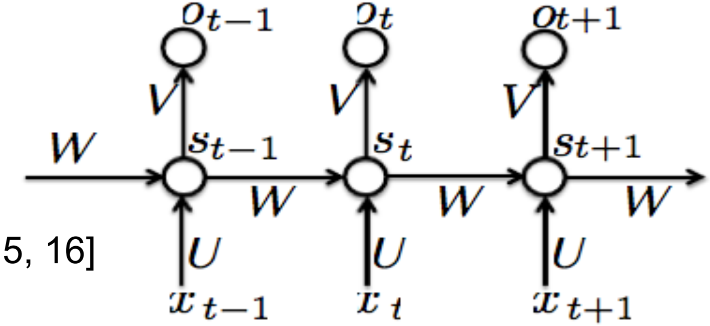
Convolution and Recurrence

- Exploit structure and prior knowledge
  - Parameter sharing is strong regularization
- Convolution - exploit locality
  - $p(y \mid X_{i-n} \ldots X_{i+n}) * p(y \mid X_{i+1-n} \ldots X_{i+1+n}) \ldots$
  - A *learned* filter over a fixed 1D or 2D window
  - Window slides over all input, updates filter
- Recurrence - exploit sequential information
  - $p(y \mid X_1 \ldots X_t) = p(y \mid X_{<=t})$ can be seen as:
  - $\sim p(y \mid X_1) * p(y \mid X_2, X_1) * p(y \mid X_3, X_2, X_1) \ldots$

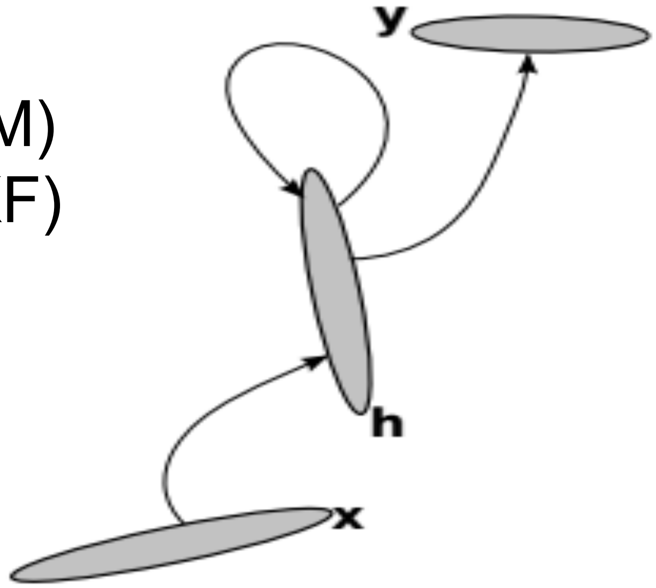
[1, 4, 5, 6, 7, 8, 9]

More on Recurrence

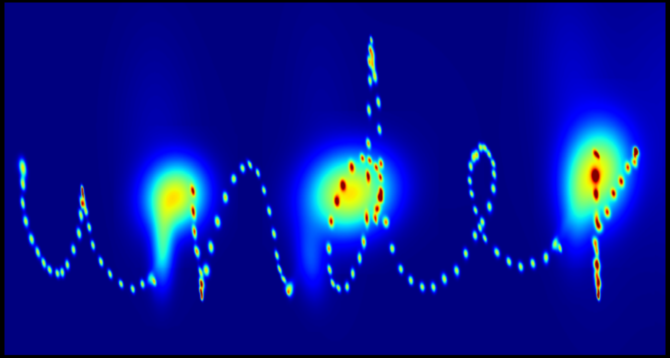
- Hidden state (s_t) encodes sequence info
  - p(X_<=t) (in s_t) is *compressed representation* of X
- Recurrence similar to
  - Hidden Markov Model (HMM)
  - Kalman Filter (KF, EKF, UKF)

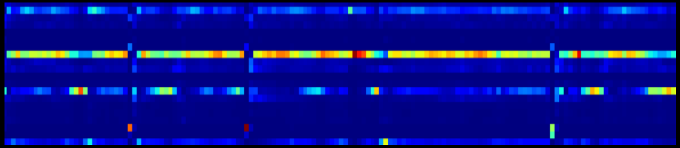


[1, 4, 15, 16]



How-To MDN + RNN



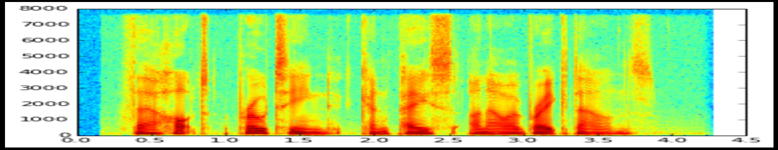


- Generating Sequences with Recurrent Neural Networks Alex Graves
  - http://arxiv.org/abs/1308.0850
- Multi-level RNN, outputs GMM and bernoulli
  - Handwriting
    - Pen up/down and relative position per timestep
  - Vocoder representation of speech
    - Voiced/unvoiced and MFCC per timestep

[3, 4]

How-To Continued

- Conditional model
  - Adds input attention (more on this later)
  - Gaussian per timestep over one hot text
  - p(bernoulli, GMM | X_t, previous state, focused text)
  - This gives *control* of the output via input text

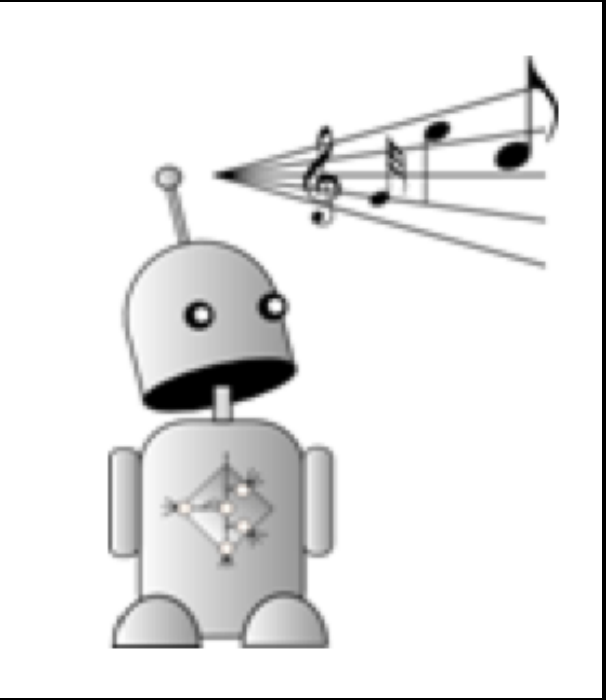


http://www.cs.toronto.edu/~graves/handwriting.html https://www.youtube.com/watch?v=-yX1SYeDHbg&t=43m30s

[3, 4]

Similar Approaches

- RNN with sigmoid output
  - ALICE
- RNN with softmax
  - RNN-LM
- RNN-RBM, RNN-NADE



[3, 1, 4, 5, 6, 7, 8, 9]

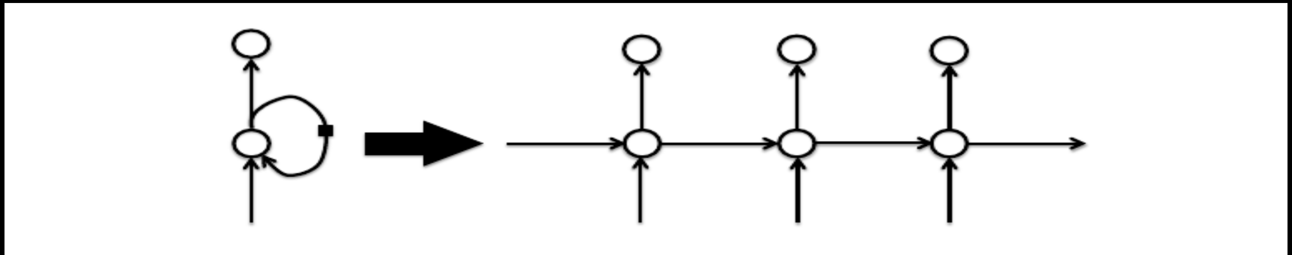
Research Questions

- Possible Issues
  - Prosody/style are not smooth over time
  - Deep network, but still shallow latent variables
  - Vocoder is a highly engineered representation
- How can we fix these problems?
  - First, a bit about conditioning in RNNs

Conditioning In Recurrent Networks

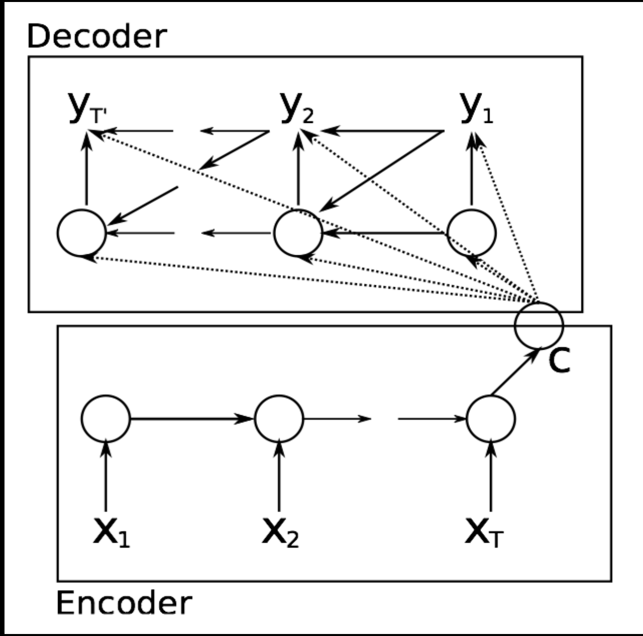
- RNNs model p(X_t | X_<t)
- Initial hidden state can condition
  - p(X_t | X_<t, c) where c is init. hidden state (context)
- Condition by concatenating in feedforward
  - Before recurrence or after
- Can do *all of the above*

[1, 4, 15, 16, 17]



Conditioning with a Sequence

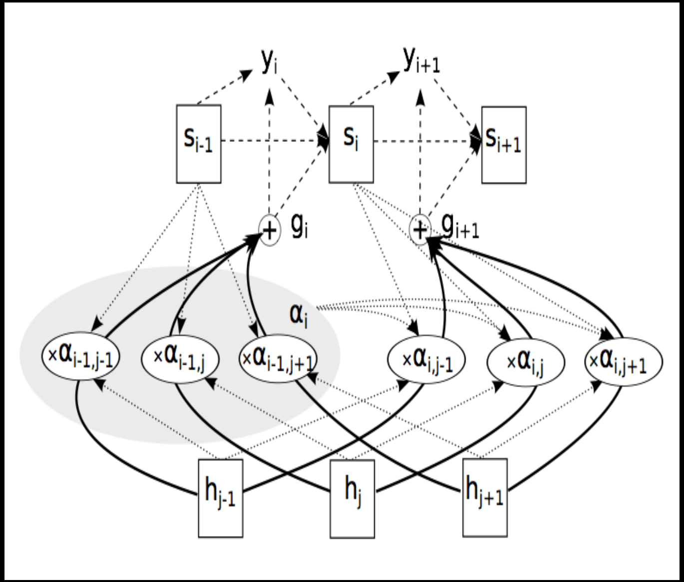
- RNN outputting Gaussian parameters over seq
  - Seen in Generating Sequences
- Use an RNN to compress
  - Hidden state encodes p(X_<=t)
  - Project into init hidden and ff
  - Now have p(y_t | y_<t, X_<=t)
  - Known as RNN Encode-Decode
  - Cho et al



[16, 17]

Distributing The Representation

- Distribute context, Bahdanau et al
- Bidirectional RNN
  - p(X_i | X_<i, X_>i) for i in t
  - Needs whole sequence
  - But sometimes this is fine
- Soft attention over hiddens
- Choose what is important

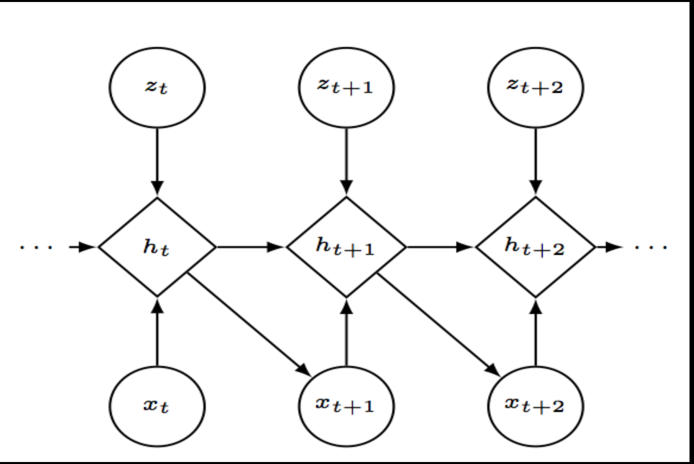


[16, 17, 18]

Previously, on FOX...

- RNN-GMM Issues
  - Prosody/style are not smooth over time
  - Deep network, but still shallow latent variables
  - Vocoder is a highly engineered representation
- How can we try to fix these problems?
  - Distributed latent representation for Z
  - Use modified VAE to make latents deep
  - Work on raw timeseries inputs
    - Extreme approach, but proves a point

Existing Approaches

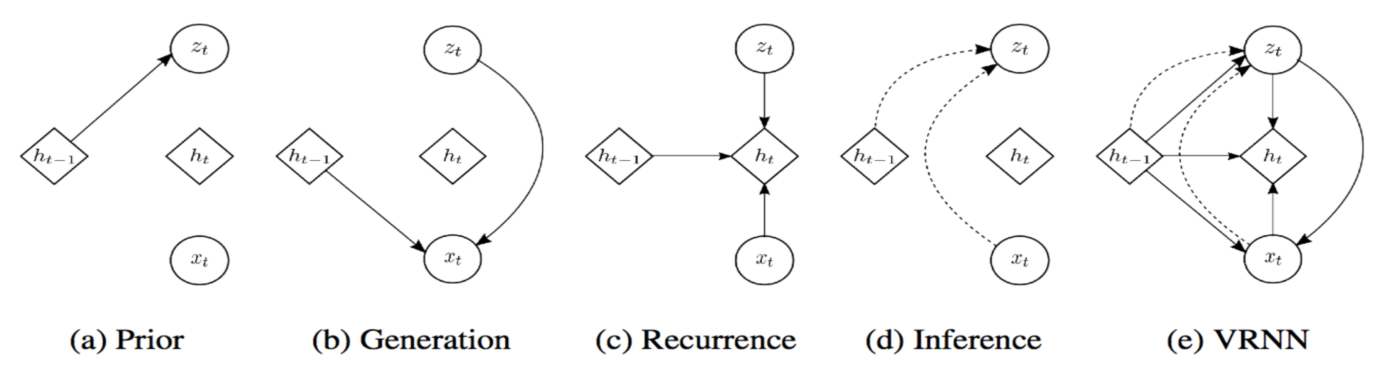


- VRAE, Z_t independent
- STORN, Z_t independent
- DRAW, Z_t loosely dependent via canvas
- No large scale real-valued experiments
  - VRAE, no real valued experiment
  - STORN, real valued experiment was small
  - DRAW, real values weren't sequences

[18, 19, 20]

Variational RNN

- Speech
  - Complex but structured noise driven by mechanics
  - Ideal latent factors include these mechanics
- Z_<t should affect Z_t and h_t
- Use a recurrent prior



[15]

Primary Functions

$$p(\mathbf{x}_{\leq T}, \mathbf{z}_{\leq T}) = \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}) p(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t}).$$

$$\mathbf{z}_t \mid \mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_{z,t}, \text{diag}(\boldsymbol{\sigma}_{z,t}^2)) \text{, where } [\boldsymbol{\mu}_{z,t}, \boldsymbol{\sigma}_{z,t}] = \varphi_\tau^{\text{enc}}(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \mathbf{h}_{t-1})$$

$$\mathbf{x}_t \mid \mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{x,t}, \text{diag}(\boldsymbol{\sigma}_{x,t}^2)) \text{, where } [\boldsymbol{\mu}_{x,t}, \boldsymbol{\sigma}_{x,t}] = \varphi_\tau^{\text{dec}}(\varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1})$$

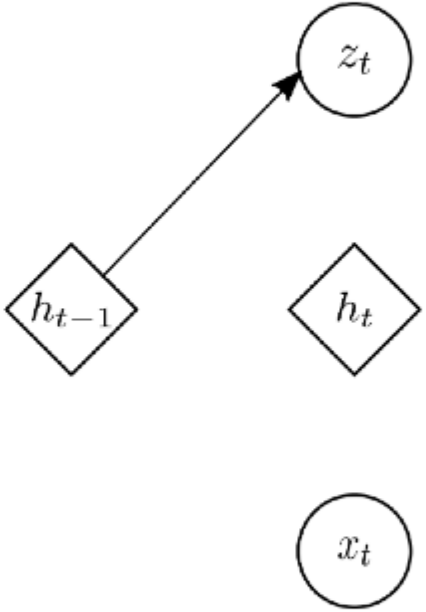
$$\mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{0,t}, \text{diag}(\boldsymbol{\sigma}_{0,t}^2)) \text{, where } [\boldsymbol{\mu}_{0,t}, \boldsymbol{\sigma}_{0,t}] = \varphi_\tau^{\text{prior}}(\mathbf{h}_{t-1})$$

$$\mathbf{h}_t = f_\theta\left(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}\right)$$

$$\sum_{t=1}^{T} -\text{KL}(q(\mathbf{z}_t \mid \mathbf{x}_{\leq t}, \mathbf{z}_{<t}) \| p(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t})) + \mathbb{E}_{q(\mathbf{z}_{\leq t} \mid \mathbf{x}_{\leq t})}\left[\log(p(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}))\right].$$

[15]

Prior



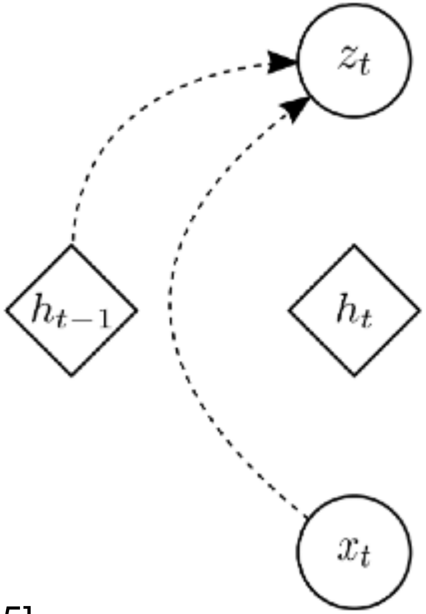
- Used for KL divergence
- Fixed in VAE to N(0, 1)
- Here it is learned
- Instead of “be simple” (as in VAE), this says “be consistent”

$$\sum_{t=1}^{T} -\mathrm{KL}(q(\mathbf{z}_t \mid \mathbf{x}_{\leq t}, \mathbf{z}_{<t}) \| p(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t}))$$
$$+\mathbb{E}_{q(\mathbf{z}_t \mid \mathbf{x}_{\leq t}, \mathbf{z}_{<t})}\left[\log(p(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}))\right].$$

[15]

$$\mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{0,t}, \mathrm{diag}(\boldsymbol{\sigma}_{0,t}^2)) \text{ , where } [\boldsymbol{\mu}_{0,t}, \boldsymbol{\sigma}_{0,t}] = \varphi_\tau^{\mathrm{prior}}(\mathbf{h}_{t-1})$$

Inference (encode)



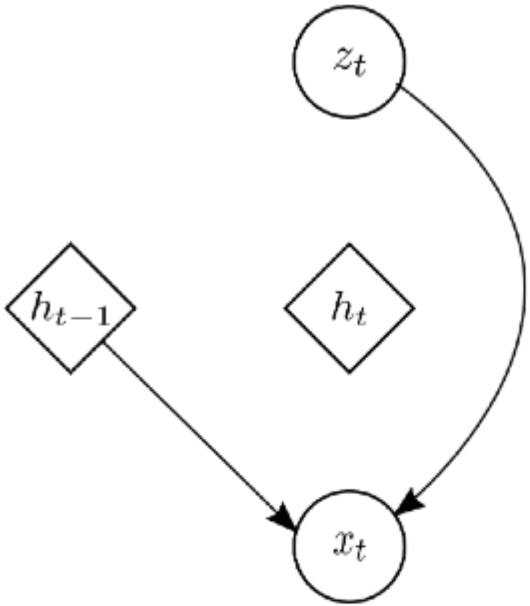
[15]

- Previous hidden state
  - h_t-1
- Data
  - X_t
- Hidden state information
  - z_<t
  - X_<t

$$\mathbf{h}_t = f_\theta\left(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}\right)$$

$$\mathbf{z}_t \mid \mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_{z,t}, \text{diag}(\boldsymbol{\sigma}_{z,t}^2)), \text{where } [\boldsymbol{\mu}_{z,t}, \boldsymbol{\sigma}_{z,t}] = \varphi_\tau^{\text{enc}}(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \mathbf{h}_{t-1})$$

Generation (decode)



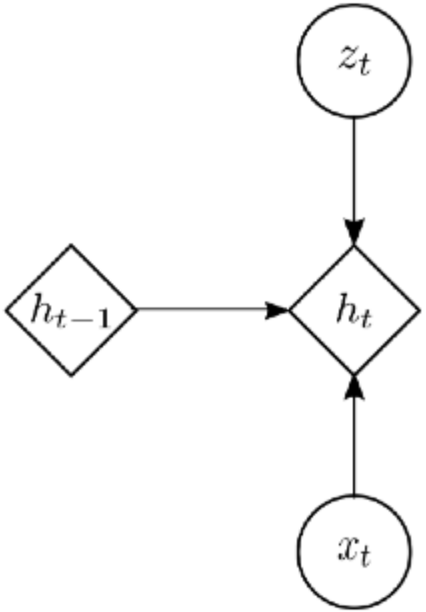
- Generate based on
  - Z_t, h_t-1
  - h_t-1 has z_<t, X_<t
  - Z_t has z_<t, X_<=t

$$\mathbf{h}_t = f_\theta\left(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}\right)$$

$$\mathbf{x}_t \mid \mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{x,t}, \text{diag}(\boldsymbol{\sigma}_{x,t}^2)), \text{ where } [\boldsymbol{\mu}_{x,t}, \boldsymbol{\sigma}_{x,t}] = \varphi_\tau^{\text{dec}}(\varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1})$$

[15]

Recurrence



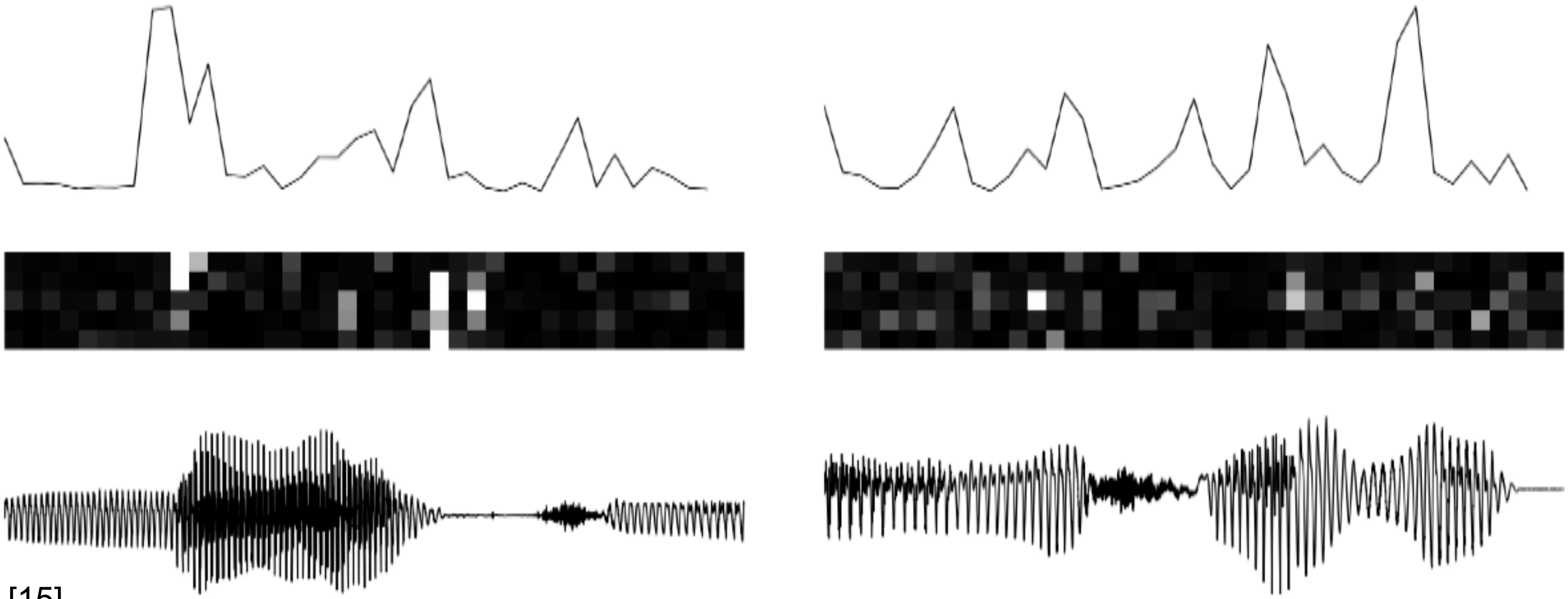
- Just a regular RNN
- Input projection is a VAE
- Can use LSTM, GRU, others

$$\mathbf{h}_t = f_\theta\left(\varphi_\tau^{\mathbf{x}}(\mathbf{x}_t), \varphi_\tau^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}\right)$$

[15]

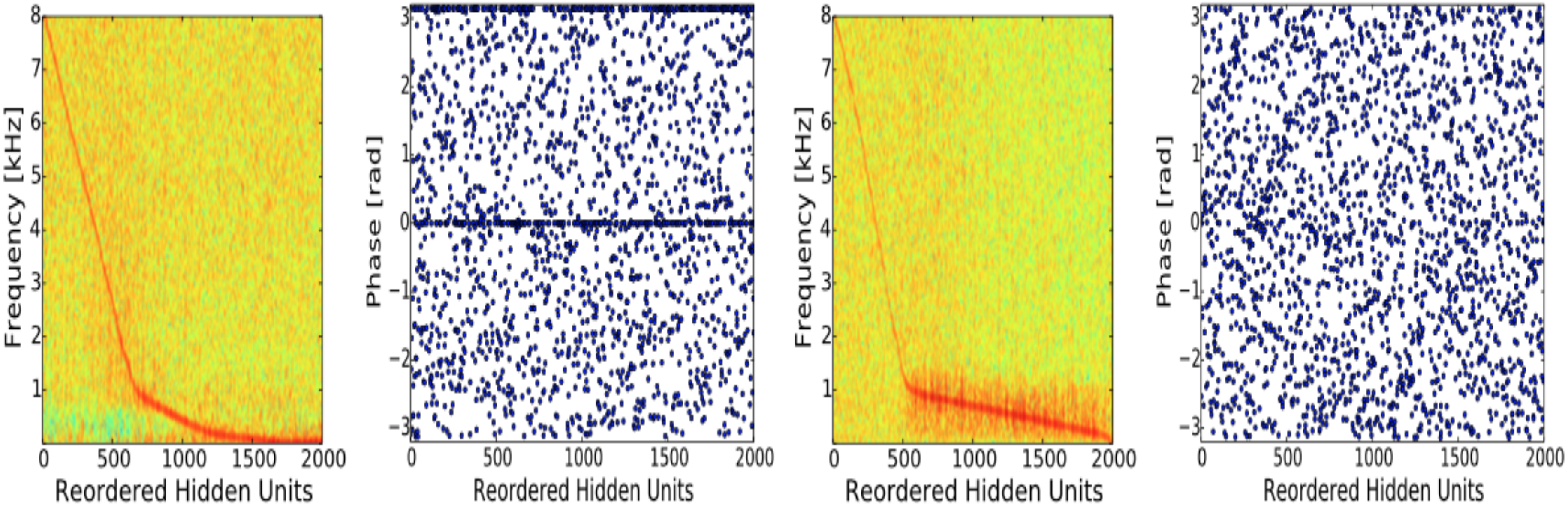
$$p(\mathbf{x}_{\leq T}, \mathbf{z}_{\leq T}) = \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}) p(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t}).$$

KL Divergence



[15]

Learned Filters



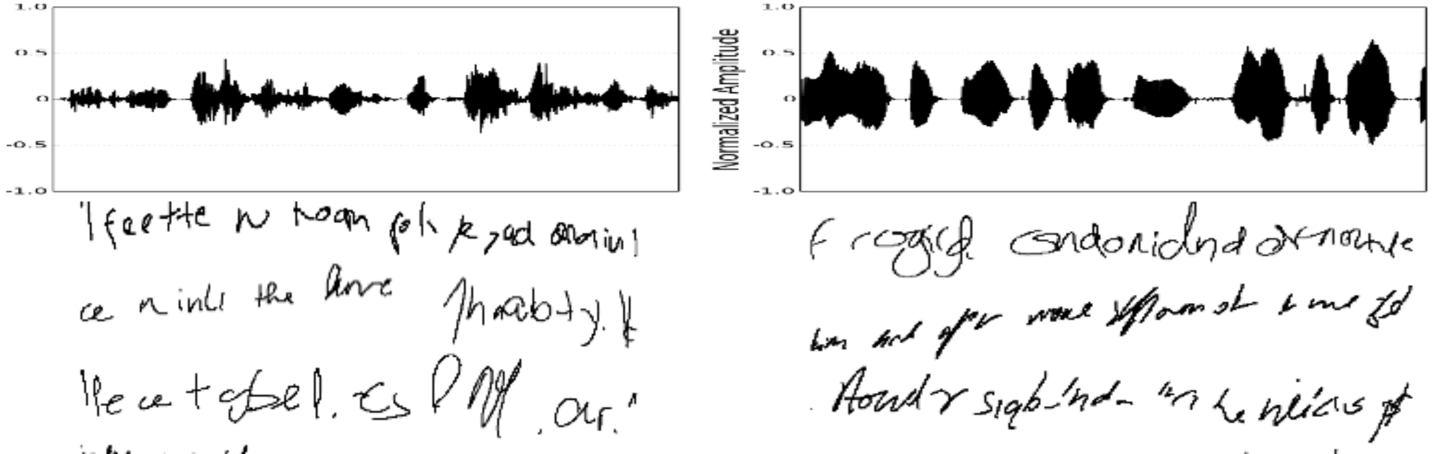
$[15]$

(a) $\varphi_{\mathcal{T}}^{\text{enc}}$ (b) $\varphi_{\mathcal{T}}^{\text{dec}}$

Final Thoughts on VRNN

- Empirically, structured Z seems to help
  - Keep style consistent
  - Predict very correlated data, like raw timeseries
  - Also works well for unconditional handwriting

RNN-GMM



VRNN-GMM

[4, 15]

Takeaways and Opinions

- Can use deep learning like graphical modeling
  - Different tools, same conceptual idea
  - Conditional probability modeling is *key*
- Put knowledge in model structure, *not* features
- Let features be *learned* from data
- Use conditioning to control or constrain

@kastnerkyle

Thanks!

Slides will be uploaded to https://speakerdeck.com/kastnerkyle

References (1)

[1] Y. Bengio, I. Goodfellow, A. Courville. “Deep Learning”, in preparation for MIT Press, 2015. http://www.iro.umontreal.ca/~bengioy/dlbook/

[2] D. Rumelhart, G. Hinton, R. Williams. "Learning representations by back-propagating errors", Nature 323 (6088): 533–536, 1986. http://www.iro.umontreal.ca/~vincentp/ift3395/lectures/backprop_old.pdf

[3] C. Bishop. “Mixture Density Networks”, 1994. http://research.microsoft.com/en-us/um/people/cmbishop/downloads/Bishop-NCRG-94-004.ps

[4] A. Graves. “Generating Sequences With Recurrent Neural Networks”, 2013. http://arxiv.org/abs/1308.0850

[5] D. Eck, J. Schmidhuber. “Finding Temporal Structure In Music: Blues Improvisation with LSTM Recurrent Networks”. Neural Networks for Signal Processing, 2002. ftp://ftp.idsia.ch/pub/juergen/2002_ieee.pdf

[6] A. Brandmaier. “ALICE: An LSTM Inspired Composition Experiment”. 2008.

[7] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, S. Khudanpur. “Recurrent Neural Network Based Language Model”. Interspeech 2010. http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf

[9] N. Boulanger-Lewandowski, Y. Bengio, P. Vincent. “Modeling Temporal Dependencies in High-Dimensional Sequences: Application To Polyphonic Music Generation and Transcription”. ICML 2012. http://www-etud.iro.umontreal.ca/~boulanni/icml2012

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, 1998. http://yann.lecun.com/exdb/mnist/

[11] D. Kingma, M. Welling. “Auto-encoding Variational Bayes”. ICLR 2014. http://arxiv.org/abs/1312.6114

[12] D. Rezende, S. Mohamed, D. Wierstra. “Stochastic Backpropagation and Approximate Inference in Deep Generative Models”. ICML 2014. http://arxiv.org/abs/1401.4082

References (2)

[13] A. Courville. “Course notes for Variational Autoencoders”. IFT6266H15. https://ift6266h15.files.wordpress.com/2015/04/20_vae.pdf

[14] D. Kingma, D. Rezende, s. Mohamed, M. Welling. “Semi-supervised Learning With Deep Generative Models”. NIPS 2014. http://arxiv.org/abs/1406.5298

[15] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, Y. Bengio. “A Stochastic Latent Variable Model for Sequential Data”. http://arxiv.org/abs/1506.02216

[16] K. Cho, B. Merrienboer, C. Gulchere, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. EMNLP 2014. http://arxiv.org/abs/1406.1078

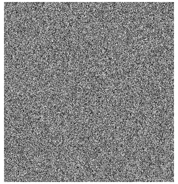
[17] D. Bahdanau, K. Cho, Y. Bengio. “Neural Machine Translation By Jointly Learning To Align and Translate”. ICLR 2015. http://arxiv.org/abs/1409.0473

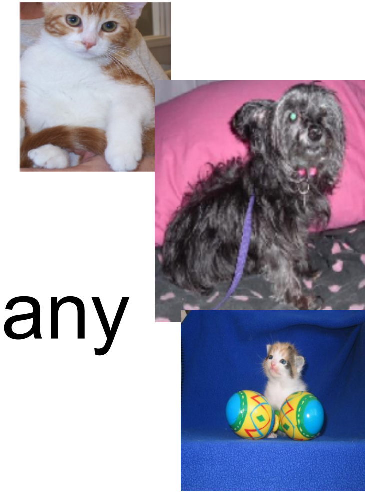
[18] K. Gregor, I. Danihelka, A. Graves, D. Rezende, D. Wierstra. “DRAW: Directed Recurrent Attention Writer”. http://arxiv.org/abs/1502.04623

[19] J. Bayer, C. Osendorfer. “Learning Stochastic Recurrent Networks”. http://arxiv.org/abs/1411.7610

[20] O. Fabius, J. van Amersmoot. “Variational Recurrent Auto-Encoders”. http://arxiv.org/abs/1412.6581

More on Convolution





- Define size of feature map and how many
  - Similar to output size of feedforward layer
- Parameter sharing
  - Small filter moves over entire input
  - Believe local statistics consistent over regions
  - Enforced by parameter sharing
- Condition by concatenating
  - Along “channel” axis
  - http://arxiv.org/abs/1406.2283