Something Old, Something New, Something Borrowed, Something Blue

Wray Buntine
Monash University
http://Bayesian-Models.org

2018-11-29
Or Thoughts On Deep Learning
From an Old Guy

⇐ ME

(before shaving)
With a Little Help From ...

<table>
<thead>
<tr>
<th>He Zhao</th>
<th>He Zhang</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ming Liu</td>
<td>Caitie Doogan</td>
</tr>
<tr>
<td>Dr. Lan Du</td>
<td>Prof. Reza Haffari</td>
</tr>
</tbody>
</table>
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
A Cultural Divide

Context: When discussing teaching Data Science with a well known professor of Statistics.

She said: “when first teaching overfitting, I always give some examples where machine learning has trouble”

I said: “funny, I do the reverse, I always give examples where statistical models have trouble”

Lesson:
We tend to have overly simple characterisations of different communities.

Lets ensure we move from Classical Machine Learning into Deep Neural Networks wisely, and not throw away the good stuff!
Motivation

I’m interested in true hybrid techniques between Classical Machine Learning and Deep Neural Networks, both theory and implementation.
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Something Old

---

Something New

---

Something Borrowed
Outline

Motivation

Examples From Classical Machine Learning
Bayesian Network Classifiers
Topic Models
Why Do They Work?

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Bayesian Network Classifiers

\[ P(A|B) = \frac{P(B|A)P(A)}{P(B)} \]
Learning Bayesian Networks

tutorial by Cussens, Malone and Yuan, *IJCAI* 2013

Bayesian Networks learning = Structure learning + Conditional Probability Table (CPT) estimation
Bayesian Network Classifiers (BNC)

- For classification or *supervised learning*.
- BNC defined by Network Structure and Conditional Probability Tables (CPTs).
- Class is $Y$ and attributes are $X_i$.
- For classification, make $Y$ a parent of all $X_i$.
- Classifies using $P(y \mid x) \propto P(y) \prod P(x_i \mid \text{parents}(x_i), Y)$

Naïve Bayes classifier: $\text{parents}(x_i) = \{y\}$

```
Y
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>X4</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>X1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>X3</td>
</tr>
</tbody>
</table>
```
k-Dependence Bayes (KDB)

Sahami, *KDD* 1996

KDB-1 classifier:
(attributes have 1 extra parent)

KDB-2 classifier:
(attributes have 2 extra parents)

**NB.** other parents also selected by mutual information
Selective k-Dependence Bayes (SKDB)
Martínez, Webb, Chen and Zaidi, *JMLR*, 2016

- SKDB is KDB where we estimate $k$ and which input variables to use.
- **Three pass learning algorithm:**
  - 1st pass, learn network structure,
  - 2nd pass, select $k$, number of parents, using LOOCV,
  - 3rd pass, learn CPTs.
- Algorithm is largely counting and sorting so is inherently scalable.

However,
- beats decision trees, but is not as good as Random Forests or Gradient Boosting of Trees$^1$

---

$^1$The top classification algorithms on Kaggle.
Improving SKDB

- Probability estimation for CPTs uses simple methods.

- There is no use of ensembles.
  - We add ensembling (Zhang, Buntine, Petitjean, forthcoming).
Why doing Hierarchical Smoothing?

- You want to predict disease as a function of some rare gene G and sex, knowing that this disease is more prevalent for females.

```
#patients with disease
#patients without disease

100-901

10-1

has gene

female

10-0

male

0-1

90-900

doesn’t have gene
```
Why doing Hierarchical Smoothing?

▶ You want to predict disease as a function of some rare gene G and sex, knowing that this disease is more prevalent for females.

\[ p(\text{disease}|\text{has-gene \& male})? \]
Why doing Hierarchical Smoothing?

- You want to predict disease as a function of some rare gene G and sex, knowing that this disease is more prevalent for females.

```
# patients with disease  # patients without disease
100–901               10–1
10–0                  90–900
```

$p_{MLE} = 0\%$
You want to predict **disease** as a function of some **rare gene G** and **sex**, knowing that this disease is more prevalent for **females**.

\[
\begin{array}{cc}
\text{has gene} & \text{doesn’t have gene} \\
100–901 & 90–900 \\
\end{array}
\]

\[
\begin{array}{cc}
\text{female} & \text{male} \\
10–1 & 0–1 \\
10–0 & 0–1 \\
\end{array}
\]

\[
p_{\text{Laplace}} = 33\%
\]
Why doing Hierarchical Smoothing?

You want to predict disease as a function of some rare gene G and sex, knowing that this disease is more prevalent for females.

\[
p_{m\text{-estimate}} = 25\%
\]
Why doing Hierarchical Smoothing?

You want to predict disease as a function of some rare gene G and sex, knowing that this disease is more prevalent for females.

<table>
<thead>
<tr>
<th>#patients with disease</th>
<th>#patients without disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>901</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>90</td>
<td>900</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

None of them use the fact that 91% of the patients with that gene have the disease!
Hierarchical Modelling

Use a hierarchical model:

\[ p(disease|\text{has-gene} \& \text{male}) : \]

leaf node, part of the model we want for inference
Hierarchical Modelling

Use a hierarchical model:

\[ p(\text{disease}|\text{has-gene} \& \text{male}) : \]

- leaf node, part of the model we want for inference

\[ p(\text{disease}|\text{has-gene}) \]

- an *abstract* parent model used to improve leaf nodes
Hierarchical Modelling

Use a hierarchical model:

\[ p(\text{disease}|\text{has-gene} \& \text{male}) : \]

leaf node, part of the model we want for inference

\[ p(\text{disease}|\text{has-gene}) \]

an abstract parent model used to improve leaf nodes

\[ p(\text{disease}) \]

an abstract grandparent model used to improve parent model
Hierarchical Modelling

Use a hierarchical model:

\[ p(\text{disease}|\text{has-gene \& male}) : \]
leaf node, part of the model we want for inference

\[ p(\text{disease}|\text{has-gene}) \]
an *abstract* parent model used to improve leaf nodes

\[ p(\text{disease}) \]
an *abstract* grandparent model used to improve parent model

**NB.** we build the hierarchies using Dirichlet distributions
Why do Ensembling?

Ensembling: we generate a set of models $\mathcal{H}$ from training data, and do inference on new case $x$ by pooling results

$$p(y|x, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{H \in \mathcal{H}} p(y|x, H)$$

- The top classification algorithms on Kaggle use ensembling

---

2Random Forests and Gradient Boosting of Trees.
Why do Ensembling?

**Ensembling:** we generate a set of models \( \mathcal{H} \) from training data, and do inference on new case \( x \) by pooling results

\[
p(y|x, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{H \in \mathcal{H}} p(y|x, H)
\]

- The top classification algorithms on Kaggle use ensembling\(^2\)
- The bias-variance-covariance decomposition of the mean square error (MSE) of ensemble \( \mathcal{H} \) (Uedo & Nakano, 1996) explains why:

\[
MSE(\mathcal{H}) = \text{bias}(\mathcal{H})^2 + \frac{1}{|\mathcal{H}|} \text{variance}(\mathcal{H}) + \left(1 - \frac{1}{|\mathcal{H}|}\right) \text{covariance}(\mathcal{H})
\]

\(^2\)Random Forests and Gradient Boosting of Trees.
Why do Ensembling?

**Ensembling:** we generate a set of models $\mathcal{H}$ from training data, and do inference on new case $x$ by pooling results

$$p(y|x, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{H \in \mathcal{H}} p(y|x, H)$$

- The top classification algorithms on Kaggle use ensembling\(^2\)
- The bias-variance-covariance decomposition of the mean square error (MSE) of ensemble $\mathcal{H}$ (Uedo & Nakano, 1996) explains why:

$$MSE(\mathcal{H}) = \text{bias}(\mathcal{H})^2 + \frac{1}{|\mathcal{H}|} \text{variance}(\mathcal{H}) + \left(1 - \frac{1}{|\mathcal{H}|}\right) \text{covariance}(\mathcal{H})$$

  i.e. larger ensemble sets with smaller covariance reduce MSE

---

\(^2\)Random Forests and Gradient Boosting of Trees.
Why do Ensembling?

**Ensembling:** we generate a set of models $\mathcal{H}$ from training data, and do inference on new case $\mathbf{x}$ by pooling results

$$p(y|\mathbf{x}, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{H \in \mathcal{H}} p(y|\mathbf{x}, H)$$

- The top classification algorithms on Kaggle use ensembling\(^2\)
- The bias-variance-covariance decomposition of the mean square error (MSE) of ensemble $\mathcal{H}$ (Uedo & Nakano, 1996) explains why:

$$MSE(\mathcal{H}) = \overline{\text{bias}(\mathcal{H})}^2 + \frac{1}{|\mathcal{H}|} \overline{\text{variance}(\mathcal{H})} + \left(1 - \frac{1}{|\mathcal{H}|}\right) \overline{\text{covariance}(\mathcal{H})}$$

  *i.e.* larger ensemble sets with smaller covariance reduce MSE

- the frequentist explanation

\(^2\)Random Forests and Gradient Boosting of Trees.
Why do Ensembling?

We want inference on new case $x$ from training data

$$
p(y|x, \text{training-data}) = \int_H p(y|x, H)p(H|\text{training-data}) \, dH$$

$$\approx \frac{1}{|\mathcal{H}|} \sum_{H \in \mathcal{H}} p(y|x, H)$$

where $\mathcal{H}$ is a representive set of models for $p(H|\text{training-data})$

- **Bayesian statistical theory** says ensembling is a good approximation to the optimal classifier (Buntine, 1989).

  i.e. since you don’t know the truth, hedge your bets with some different options

- the frequentist and Bayesian approaches have great similarity!
Improved SKDB

- With hierarchical smoothing, a *single* SKDB beats Random Forests in MSE and 0-1 loss, and is more scalable.
  - Smoothed SKDB $\gg$ Random Forests

- With hierarchical smoothing, an ensemble of SKDB beats Gradient Boosting of Trees in MSE and 0-1 loss, and is similar in speed.
  - Smoothed Ensembled SKDB $\gg$ Gradient Boosting of Trees

for discrete data, ..., currently
Outline

Motivation

Examples From Classical Machine Learning
   Bayesian Network Classifiers
   Topic Models
   Why Do They Work?

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Topic Models

from http://bayesian-models.org
Latent Dirichlet Allocation

Blei, Ng, Jordan *JMLR* 2003

Topics

gene 0.04
dna 0.02
genetic 0.01
...

life 0.02
evolve 0.01
organism 0.01
...

brain 0.04
neuron 0.02
nerve 0.01
...

data 0.02
number 0.02
computer 0.01
...

Documents

**Seeking Life’s Bare (Genetic) Necessities**

Cold Spring Harbor, New York—How many genes does an organism need to survive? Last week at the genome meeting here, two genome researchers with radically different approaches presented complementary views of the basic genes needed for life.

One research team, using computer analysis to compare known genomes, concluded that today’s organism can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 would not be enough.

Although the numbers don’t match precisely, those predictions are not all that far apart,” especially in comparison to the 75,000 genes in the human genome, notes Siv Anderson, an assistant professor at the University of Washington. In Sweden, she has arrived at the 80 number. But coming up with a consensus answer may be more than just a matter of numbers. “In particular, as more and more genomes are completely mapped and sequenced,” explains Araceli Mushakian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing all...
Matrix Approximation

\[ W \approx \Theta \Phi^T \]

<table>
<thead>
<tr>
<th>Data $W$</th>
<th>Components $\Theta$</th>
<th>Error</th>
<th>Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>real valued</td>
<td>unconstrained</td>
<td>least squares</td>
<td>PCA and LSA</td>
</tr>
<tr>
<td>non-negative</td>
<td>non-negative rates</td>
<td>least squares</td>
<td>NMF, learning codebooks</td>
</tr>
<tr>
<td>non-neg int.</td>
<td>probabilities independent</td>
<td>cross-entropy</td>
<td>Poisson &amp; Neg.Bino. MF</td>
</tr>
<tr>
<td>non-neg int.</td>
<td>scores</td>
<td>cross-entropy small</td>
<td>topic models</td>
</tr>
<tr>
<td>real valued</td>
<td>shifted PMI</td>
<td></td>
<td>ICA</td>
</tr>
<tr>
<td>non-neg int.</td>
<td></td>
<td></td>
<td>GloVe</td>
</tr>
</tbody>
</table>
Matrix Approximation Terminology

- **Statistics**: “components”
- **Classical ML**: “topics”
- **Deep NNs**: “embeddings”
Component Models, Generally

Approximate faces/bag-of-words (RHS) with a linear combination of components (LHS).
Improving Topic Models: I

Different topics should have different base rates.

▶ Consider the following topics in news about “Obesity”:

▶ say have obesity not health need problem issue  
  → 10.7% of words
▶ christ religious faith jewish bless wesleyan  
  → 0.08% of words

▶ Standard LDA says these two should be equally likely.
Improving Topic Models: I

Different topics should have different base rates.

- we make priors on the topic proportions asymmetric,
- done by Teh, Jordan, Beal and Blei 2006
  - spawned Hierarchical Dirichlet processes (HDP) and nested/hierarchical Chinese restaurants
Different topics should have different base rates.

- we make priors on the topic proportions asymmetric,
- done by Teh, Jordan, Beal and Blei 2006
  - spawned Hierarchical Dirichlet processes (HDP) and nested/hierarchical Chinese restaurants
- done by Wallach, Mimno, McCallum 2009
  - now available in the Mallet topic modelling system
Improving Topic Models: I

Different topics should have different base rates.

- we make **priors on the topic proportions asymmetric**, done by Teh, Jordan, Beal and Blei 2006
  - spawned **Hierarchical Dirichlet processes (HDP)** and nested/hierarchical Chinese restaurants
- done by Wallach, Mimno, McCallum 2009
  - now available in the Mallet topic modelling system
- considerable theory and algorithms, 2009-2012
  - notable mention: Bryant and Sudderth, 2012
  - but some implementations gave poor results
Improving Topic Models: I

Different topics should have different base rates.

- we make priors on the topic proportions asymmetric,
- done by Teh, Jordan, Beal and Blei 2006
  - spawned Hierarchical Dirichlet processes (HDP) and nested/hierarchical Chinese restaurants
- done by Wallach, Mimno, McCallum 2009
  - now available in the Mallet topic modelling system
- considerable theory and algorithms, 2009-2012
  - notable mention: Bryant and Sudderth, 2012
  - but some implementations gave poor results
- done by Buntine and Mishra, KDD, 2014
  - does HDP efficiently with a fast Gibbs sampler
  - multi-core, great results
  - Gibbs sampling beats variational inference!
Yields High Fidelity Topics

Examples from 100 topics about “Obesity in the ABC news” from 2003-2012, from 600 news articles of average length 150 words:

<table>
<thead>
<tr>
<th>rank</th>
<th>percentage</th>
<th>words</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4.57%</td>
<td>study researcher finding journal publish twice university</td>
</tr>
<tr>
<td>14</td>
<td>1.54%</td>
<td>teenager boy child adults parent youngster bauer school-child</td>
</tr>
<tr>
<td>22</td>
<td>0.86%</td>
<td>doctor ambulance hospital psychiatric general-practitioner staff</td>
</tr>
<tr>
<td>42</td>
<td>0.43%</td>
<td>soft-drink instant soda carbonated fizzy beverages candy sugary</td>
</tr>
<tr>
<td>78</td>
<td>0.18%</td>
<td>olympics time second olympic pool win team freestyle gold</td>
</tr>
<tr>
<td>91</td>
<td>0.11%</td>
<td>colonel lieutenant-general afghanistan rifle stirling mission</td>
</tr>
<tr>
<td>95</td>
<td>0.10%</td>
<td>dialysis end-stage dementia kidney-disease kidney abdominal</td>
</tr>
</tbody>
</table>

- 100 topics for 600 documents
- most are on coherent subjects
Improving Topic Models: II

Words in text are bursty: they appear in small bursts.

Original news article:
Women may only account for 11% of all Lok-Sabha MPs but they fared better when it came to representation in the Cabinet. Six women were sworn in as senior ministers on Monday, accounting for 25% of the Cabinet. ...

Bag of words:

11% 25% Cabinet(2) Lok-Sabha MPs Monday Six They
Women account accounting all and as better but came fared for(2) in(2) it may ministers of on only representation senior sworn the(2) to were when women

► effect is called burstiness
► first modelled by Doyle and Elkan 2009, but intolerably slow
► done by Buntine and Mishra, KDD, 2014 using HDPs
  ► only 25% (or so) penalty in memory and time
  ► huge improvement in perplexity, and smaller one in coherence
  ► but loss of fidelity (“fine” low probability topics)
  ► so we usually don’t use
Improving Topic Models: III

Information about word similarity/semantics should be used when building topics.

- we use prior information about words from embeddings
- done recently by many in topic modelling and deep neural networks

from “An Introduction to Word Embeddings”, blog by Roger Huang, 2017
ASIDE: Multi-Label Learning (MLL)

- same source data
- multiple labels
- one combined model/system to do it

Is the article written in Mandarin?
Is the article about President Trump?
Is the article about economics?
Does the article have positive sentiment?
ASIDE: Multi-Task Learning (MTL)

- different source data
- different labels or tasks
- one combined model/system to do it
ASIDE: Naive Multi-Task Learning

Have $T$ somewhat related separate classification tasks. Predict $Y_t$ from $X_t$ using parameters $\Theta_t$.

$$p(Y_t|X_t, \Theta_t) \quad \text{for } t = 1, \ldots, T$$
ASIDE: Multi-Task Learning (MTL)

Add a shared parameter $\Theta^G$ which captures “common knowledge”.

\[
p(\tilde{\Theta}_t | \Theta^G) \quad \text{for } t = 1, \ldots, T
\]

\[
p(Y_t | X_t, \Theta_t, \tilde{\Theta}_t) \quad \text{for } t = 1, \ldots, T
\]

NB. another hierarchical model with $\Theta^G$ the parent node
Prior Regression for MTL

Regress from metadata $C_t$ onto task-specific version of common knowledge $\tilde{\Theta}_t$, using parameters $\Theta^G$.

$$p(\tilde{\Theta}_t | C_t, \Theta^G) \quad \text{for } t = 1, \ldots, T$$

$$p(Y_t | X_t, \Theta_t, \tilde{\Theta}_t) \quad \text{for } t = 1, \ldots, T$$

**NB.** in statistics, random effects models achieve this effect
Improving Topic Models: III

Information about word similarity/semantics should be used when building topics.

- we use prior information about words from embeddings
- done recently by many in topic modelling and deep neural networks
- done using prior regression by Zhao, Du, Buntine, Liu *ICDM* 2017, Zhao, Du, Buntine, *ACML* 2017
  - regress the metadata (e.g., word embeddings, document labels) onto the model parameters during learning
  - using fast “gamma regression”
- code available at He Zhao’s GitHub repo
- very good results
Improving Topic Models: IV

Hierarchical structure between topics should be discovered.

- once we go beyond 20 topics, this supports explanation
Topics Enhanced with Word Embeddings
Zhao, Du, Buntine, Zhou ICML 2018

these are the regular topics as per LDA
Topics Enhanced with Word Embeddings
Zhao, Du, Buntine, Zhou ICML 2018

this is a "sub-topic", formed with the help of embeddings
Topics Enhanced with Word Embeddings
Zhao, Du, Buntine, Zhou ICML 2018

2. xbox microsoft console launch game playstation day friday sale sony

3. friday black thanksgiving shopping deal day holiday store year monday

4. xbox microsoft console launch game playstation sony kinect sale sold

1. year time thing don good long well big will day

thing think bit pretty love good going feel way great

time year long well ago thing good lot will thought
donwhen doesn shouldn first would could going
don players game games storyline how will works

3. topic

10. company best video sold year edition tour early microsoft

xbox microsoft console kinect gaming playstation system apple will nintendo

several sub-topics combine to form single topic
Topics Enhanced with Word Embeddings
Zhao, Du, Buntine, Zhou ICML 2018

topics themselves formed into a hierarchy

Intra-topic structure
Outline

Motivation

Examples From Classical Machine Learning
  Bayesian Network Classifiers
  Topic Models
  Why Do They Work?

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Why Do They Work?
Why Do They Work?

Classification with Smoothed, Ensembled BNCs:

▶ partitioning (sorting and counting)
  ➞ computation is scalable

▶ hierarchical models and smoothing
  ➞ helps prevent overfitting on single model

▶ ensembles
  ➞ giving us great learning performance since 1988!
Why Do They Work?

Topic Models with Rich Priors and Structures:

- prior regression
  - uses metadata so parameters for similar items will end up being similar
- hierarchical ("deep") Bayesian models
  - like deep neural networks, they learn shared structures
- Gibbs sampling
  - a generic estimation tool we can automate, and can be done efficiently with multicore or GPUs
Something New
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks
  Neural Machine Translation

Active Learning and Other Methods

Representation Theory

Why Do They Work?

Moving Forward

Some Reflections

Conclusion
Neural Machine Translation
Bilingually low-resource scenario: large amounts of bilingual training data is not available.

**IDEA:** Use existing resources from other tasks and train one model for all tasks using multi-task learning (MTL).
Add three additional tasks after the primary translation task.

Machine Translation

I went home

 Semantic Parsing

Obama was elected and his voter celebrated

 Syntactic Parsing

The burglar robbed the apartment

 Named-Entity Recognition

Jim bought 300 shares of Acme Corp. in 2006
NMT: Basic Setup

Train on the 4 tasks with a task indicator.
Reminder: Multi-Task Learning (MTL)

Use the standard MTL setup.
Extend a standard recurrent neural network model by adding multi-tasking blocks and a gating controller.
Block-1 to Block-3 are task independent components, $\Theta^G$ the shared common knowledge for MTL

Routing-Network controls their use on a task to create $\tilde{\Theta}_t$

task specific parameter is $\Theta_t$
NMT: Results

- Implementation for the RNN uses 400 hidden states.
- Experiments with English to Farsi and English to Vietnamese (about 100k sentence pairs each in training).
- Good improvements in BLUE and Perplexity over other methods.
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Neural Machine Translation

Active Learning and Other Methods

Representation Theory

Why Do They Work?

Moving Forward

Some Reflections

Conclusion
Active Learning

Category model

“bicycle”

flickr

Annotated data

Unlabeled images

Selected examples

(from kisspng.com “active learning machine learning”)
Active Learning

Category model

"bicycle"

flickr

Annotated data

Unlabeled images

Selected examples

(from kisspng.com "active learning machine learning")
Active learning is a useful technique when labelled data is inadequate for classification.

Various heuristics exist to propose new instances for the Oracle/Expert to label:
- uncertainty sampling
- diversity sampling
- random sampling

**IDEA:** Use pool of related problems with available labelled data and train a “tutor” to suggest instances.

- uses reinforcement learning
- technique is called imitation learning
  - Ross & Bagnell, 2014
Other Methods
What other variants of the MTL template are there?

- learn to initialise parameters values
- learn SGD hyper-parameters, learning rate, etc.

E.g.  
- Model-agnostic meta-learning, Finn et al. 2017
- Meta-SGD, Li et al. 2017
Notable Mentions

  - documents have a hierarchical structure
  - model attention to do classification
  - great classification results

  - straight forward NN with hidden layer
  - full sequence modelling, not bag-of-words
  - great predictive results (we checked)

- several papers at ACML and workshops
- many more!
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks
  Neural Machine Translation
  Active Learning and Other Methods
  Representation Theory
  Why Do They Work?

Moving Forward

Some Reflections

Conclusion
Representation Theory
Main Idea: if we use a “simpler” class of models, then learning must happen faster, but the resultant learned model may not be as good.

e.g. class of polynomials of degree at most $n$,

- Various versions of theory: VC dimension, Rademacher complexity, uniform stability.
ASIDE: Regularisation Theory

**Main Idea:** Add a complexity measure to the error term and optimise a multi-objective function:

\[ \text{model-error} + \lambda \cdot \text{model-complexity} \]

for different \( \lambda \).

- An old idea, developed by mathematicians in 1970’s as solution to ill-posed problem.
- Independently developed as minimum description length (MDL) and minimum message length (MML) in the 1960-70’s too.
- Has a Bayesian interpretation.
MSE for linear models with basis functions with $p$ parameters and $N$ data with $d$ dimensions, cannot do better than

$$O \left( \frac{1}{p^{2/d}} \right) + O \left( \frac{p}{N \log N} \right)$$

MSE for 2-layer neural nets with sigmoidal units with $r$ nodes and $N$ data with $d$ dimensions (so $p = O(rd)$ parameters) is

$$O \left( \frac{1}{r} \right) + O \left( \frac{p}{N \log N} \right)$$
deep neural networks improve over standard capacity and regularisation theory

many similar results, e.g., discussion in Zhang, Bengio, Hardt, Recht, Vinyals *ICLR* 2017

deep networks really are special, they learn better with same number of parameters
  Yann LeCunn always said this, based on empirical evidence
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks
  Neural Machine Translation
  Active Learning and Other Methods
  Representation Theory
  Why Do They Work?

Moving Forward

Some Reflections

Conclusion
Why Do They Work?
Why Do They Work?

- Model/Spec driven black-box algorithms ease the work load of developers.
  - machine learning without statistics!
- Porting down to GPUs or multi-core allows real speed.
- Deep models allow more effective learning and higher order concepts to be discovered
  - convolutions, structures, sequences, ...
  - so-called representation learning
- High capacity makes them very flexible in fitting.
- Allows “modelling in the large”:
  - learning to learning
  - multi-task learning
  - imitation learning
  - convolutions, structures, sequences, ...
The Old Versus The New: I

The Old: need experts to carefully design algorithms:
- experts need knowledge of distributions and techniques like variational algorithms or Gibbs samplers to construct algorithms
- statistical knowledge intensive

The New: (semi) automatic black-box algorithms:
- automatic differentiation, ADAM optimisation, etc.
- port down to GPUs or multi-core, etc.
- easier to scale algorithms
The Old Versus The New: II

The Old: modelling in the small:
- huge range of components can be used
- individual components need care and attention for algorithm development

The New: modelling in the large:
- whole blocks can be composed
- general purpose methods deal with it
- restricted in allowable components
  - use concrete distribution and reparameterisation trick
The Old Versus The New: III

The Old: components often directly interpretable:
★ parameter vectors can have easy interpretation

The New: black-box model requires “explanation” support:
★ cannot interpret the model
★ need techniques like LIME and SHAP to interpret results
The Old Versus The New: Impact

**The New:** allows a huge expansion in capability.
- automatic black-box algorithms
- learning to learn
- modelling in the large
  - e.g. porting to special purpose hardware

**The New:** but there is some loss.
- interpretable models
- whole classes of algorithms
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Automating Statistical Inference

from Buntine JAIR 1994
Modelling language:

```r
model{
  # model priors
  beta0 ~ dnorm(0, 0.001)
  eta1 ~ dnorm(0, 0.001)
  tau ~ dgamma(0.1, 0.1)
  sigma <- 1/sqrt(tau)
  # data model, linear regression
  for( i in 1:n) {
    mu[i] <- beta0 + beta1*x[i]
    y[i] ~ dnorm(mu[i] , tau)
  }
}
```

- Simple Bayesian linear regression using Gaussian model \( \tilde{x} = \beta_0 + \beta_1 \tilde{y} \).
- All constants, parameters and data are defined in the language.
Modelling language using Bayesian networks to specify probability models.
  ▶ compiles to stack-based intermediate code (like Java)

 Runs a simulation on the network to generate a set of typical variable values, i.e., a sample.
  ▶ runs a Gibbs sampler

 Revolutionised the application of statistics in mid 90’s.
Stan: similar to BUGS language but uses Hamiltonian Monte Carlo (HMC); from Columbia

TFP: TensorFlow Probability (TFP), combines probabilistic models and deep learning on modern hardware
  ▶ from the TensorFlow team at Google, released April 2018

Edward: broad variety of statistical learning, in Python on TensorFlow
  ▶ http://edwardlib.org/ by Dustin Tran in TFP group, ex Blei student

Greta: simple and scalable statistical modelling in R, built on Google’s TensorFlow
  ▶ Nick Golding, on GitHub, 2018
These efforts have related goals to deep neural network modelling.

- network modelling language
- general inference routines

Consequently, had a huge impact within applied statistics.

Limited support for discrete data, and model transformations.

Mixed ability to scale up.

- OK for smaller scale statistical experimentation.
- but they’re starting to scale-up ... (e.g., Greta)
Automating Statistical Operations

Sachith and Buntine, 2019 (in progress)

```java
//Initialization
for (int m = 0; m < M; m++){
    for (int n = 0; n < N; n++){
        z[m][n]=Math.Random()*K;
        c0[m][z[m][n]]++;
        c0_1[m]++;
        c1[z[m][n]][w[m][n]]++;
        c1_1[z[m][n]]++;
    }
}
//For each iteration of the Markov Chain run the following:
for (int m = 0; m < M; m++){
    for (int n = 0; n < N; n++){
        c0[m][z[m][n]]--;
        c0_1[m]--;
        c1[z[m][n]][w[m][n]]--;
        c1_1[z[m][n]]--;
        //Sample from full conditional
        double[] p = new double[K];
        for (int k = 0; k < K; k++){
            p[k]=(Math.pow(a_1+c0_1[m],-1))*(Math.pow(b_1+
        }
    //cumulate values
    for (int k = 1; k < K; k++){
        p[k]+=p[k-1];
    }
    int k;
    double val = Math.random()*p[K-1];
```

- most approaches use general schemes
- at Monash we’re automating statistical operations and fast Gibbs samplers
- focussing on discrete models
- able to generate optimised/specialised samplers
- able to port down to multicore

(optimised Gibbs sampler for LDA)
Automating Statistical Inference, cont.

We need to **borrow** from the statistical “automation” efforts and combine them with deep neural networks.

This is how we make deep neural networks more probabilistic.
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Something Blue
Our Experiments with Deep Topic Models

Our comparison:

- evaluate perplexity using last model found: 
  \[ p(\text{new-doc}|\text{data}, \hat{\text{model}}) \]
- a quick comparison: other small datasets, used 100 topics
- using related code we could get our hands on

<table>
<thead>
<tr>
<th>(method)</th>
<th>20NG</th>
<th>WS</th>
<th>TMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVLDA</td>
<td>1240</td>
<td>3186</td>
<td>5137</td>
</tr>
<tr>
<td>PRODLDA</td>
<td>1226</td>
<td>2997</td>
<td>5041</td>
</tr>
<tr>
<td>NVDM (last)</td>
<td>2085</td>
<td>4647</td>
<td>6086</td>
</tr>
<tr>
<td>NVDM (best)</td>
<td>1322</td>
<td>2311</td>
<td>3804</td>
</tr>
<tr>
<td>LDA-standard</td>
<td>781</td>
<td>983</td>
<td>2026</td>
</tr>
<tr>
<td>MetaLDA (ours)</td>
<td>763</td>
<td>944</td>
<td>1891</td>
</tr>
</tbody>
</table>

+ burstiness: another -100 to -300!

DocNADE: lower again!
Discussion

- Some deep learning methods aren’t performing well against other methods.
  - oftentimes compared against poor quality variants
  - for perplexity and topic coherence

- But some deep neural network models work very well:
  - DocNADE (Larochelle & Lauly, NIPS 2012) substantially beats LDA (we tested it).
  - LSTM (Zaheer, Ahmed & Smola, ICML 2017) substantially beats LDA (has stronger empirical work).
  - Both are sequential models.
Experiments with Deep Topic Models

**Claim:** Better empirical work is needed. The deep neural network models aren’t always better.

**Claim:** An underlying problem is an information deluge in the machine learning community!

**NB.** too many conferences and journals ... hard for even the best to stay on top of all work
Outline

Motivation

Examples From Classical Machine Learning

Examples From Deep Neural Networks

Moving Forward

Some Reflections

Conclusion
Conclusion

▶ The Old (classical machine learning) now an advanced state:
  ▶ ensembles, deep models, regularising, Bayesian inference
  ▶ a degree of automation starting (JAGS, Stan)

▶ The New (deep neural networks) works well, but not always.
  ▶ limited in probabilistic methods
Conclusion: Claim 1

The success of deep neural networks is not due intrinsically to neural networks.

- it is compiling down to GPUs
- it is ADAM and general purpose inference
- it is learning “in the large”
- it is “deep” models
- it is the influx of creativity
Conclusion: Claim 2

Probability theory plus Optimisation is the general “theory of learning.”

- everything else is just special cases
- deep neural nets still has all the same aspects to consider:
  - capacity, regularisation, ...
  - overfitting, ensembles, ...
  - subjectivity, objectivity, belief, ...
Conclusion: Claim 3

The next frontier in learning is adding back the old ML techniques and integrating new general statistical inference into the new computational frameworks.

▶ Google agrees:
  ▶ building TensorFlow Probability
▶ Nvidia agrees:
  ▶ they want to broaden applications beyond deep neural networks
▶ HMC samplers already done (i.e., Stan)
▶ starting work for variational inference (Edward)
▶ ...

Questions?
Probabilistic Modelling in Learning


- “full” probabilistic modelling is Bayesian modelling
- probability theory is the only coherent theory of uncertain reasoning
- concepts such as “Capacity” and “Regularisation” are important
  - no doubt there are more
- deep neural networks provide a new computational paradigm, but doesn’t change theory of learning