Outline

Part I:
1. Intro, motivation
2. Machine learning 101
3. Neural nets, backprop, RNNs
4. Applications

Part II:
1. Structured prediction
2. Unsupervised learning
3. Attention → Reasoning → “Neural programs”
4. Architecture exploration
5. Towards hardware-friendlier DL
6. Software
Rosenblatt’s perceptron (1957)

pictures from http://www.rutherfordjournal.org/article040101.html
Rosenblatt’s perceptron (1957)

\[ y(\sum_i w_i x_i) = y(w^T x) \]

▶ “the embryo of an electronic computer that (the Navy) expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence”

(in NYT according to wikipedia)
Rosenblatt’s perceptron (1957)

\[ y(\sum_i w_i x_i) = y(w^T x) \]

- “the embryo of an electronic computer that (the Navy) expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence”

(in NYT according to wikipedia)
ML allows us to harness training data \((x_n, t_n)\) \(n = 1 \ldots N\)
ML allows us to harness training data \((x_n, t_n)_{n=1\ldots N}\)
ML allows us to harness training data $(x_n, t_n)_{n=1...N}$

- ML allows us to harness parallelization
Machine Learning

\[ w^T x > 0 \]

\[ w^T x = 0 \]

\[ w^T x < 0 \]
The XOR problem
The XOR problem and multi-stage processing
The XOR problem and multi-stage processing
Multi-stage processing

\[ x \rightarrow W^{01} \rightarrow h_1 \rightarrow W^{12} \rightarrow h_2 \rightarrow W^{23} \rightarrow h_3 \rightarrow W_{out} \rightarrow y \]
“It’s the features, stupid!”
“It’s the features, stupid!”
“It’s the features, stupid!”

A common computer vision pipeline before 2012

1. Find interest points.
“It’s the features, stupid!”

A common computer vision pipeline before 2012

1. Find interest points.
2. Crop patches around them.
“It’s the features, stupid!”

A common computer vision pipeline before 2012

1. Find interest points.
2. Crop patches around them.
3. Represent each patch with a sparse local descriptor.
“It’s the features, stupid!”

\[
\begin{pmatrix}
  f_1^1 \\
  \vdots \\
  f_n^1 \\
  \vdots \\
  f_1^M \\
  \vdots \\
  f_n^M
\end{pmatrix}
\]

A common computer vision pipeline before 2012

1. Find interest points.
2. Crop patches around them.
3. Represent each patch with a sparse local descriptor.
4. Combine the descriptors into a representation of the image.
“It’s the features, stupid!”

This creates a representation that even a linear classifier can deal with.
“It’s the features, stupid!”

This creates a representation that even a linear classifier can deal with.

bottom line: **non-linear pipelines are useful**
(aka “the representation matters”)
What do good low-level features look like?

▷ Local features that are often found to work well are based on oriented structure (such as Gabor features).

▷ These were discovered again and again (also in other areas) and are closely related to the Short Time Fourier Transform.
Neural networks are \textit{trainable} pipelines.
Neural networks are \textit{trainable} pipelines

Most common networks interleave \textbf{matrix multiplies} with \textbf{element-wise non-linearities}:

\[
y(x) = W^{\text{out}} h(W^{23} h(W^{12} h(W^{01} x)))
\]

Usually there are constant “bias”-terms as well.
Neural networks are trainable pipelines

Common non-linearities:
- sigmoid: \( h(a) = \frac{1}{1 + \exp(a)} \)
- ReLU: \( h(a) = a \cdot [a > 0] \)
- tanh: \( h(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}} \)
Neural networks are *trainable* pipelines

For classification tasks, turn class outputs into probabilities using the "softargmax" function:

\[
p(C_k | x) = \frac{\exp(y_k(x))}{\sum_j \exp(y_j(x))}
\]
Neural networks are trainable pipelines

For training, use a (large) training set \((x_n, t_n)_{n=1\ldots N}\) and minimize a suitable cost-function. The minimization is usually done using stochastic gradient descent (SGD).
The most common choices of cost function

- **Regression** (predict real values):
  \[
  \text{cost} = \frac{1}{2} \sum_{n=1}^{N} \| y(x_n) - t_n \|^2
  \]

- **Classification** (predict discrete labels):
  \[
  \text{cost} = - \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log p(C_k | x_n)
  \]
  where \( t_{nk} = 1 \) iff training case \( n \) belongs to class \( k \).
Stochastic gradient descent (SGD)

\[ \theta^{(\tau+1)} = \theta^{(\tau)} - \eta \frac{\partial \text{cost}(x_n, t_n)}{\partial \theta} \]

For one or several training cases at a time, iterate:
1. compute cost (forward pass)
2. compute derivatives (backward pass)
3. update parameters
Stochastic gradient descent (SGD)

\[
\theta(\tau+1) = \theta(\tau) - \eta \frac{\partial \text{cost}(x_n, t_n)}{\partial \theta}
\]

- Most operations performed on each training example will be matrix-vector products.
- To get a higher arithmetic intensity it is common to use **mini-batches** (often of size \(\approx 100\), currently...).
- Each full pass through the training set is called an **epoch**.

▶ **Use the chainrule!** For regression and classification we get:

\[
\frac{\partial \text{cost}}{\partial y(x_n)} = y(x_n) - t_n
\]

▶ Next: If \( y \) has any parameters, \( W^{\text{out}} \), collect them using:

\[
\frac{\partial \text{cost}}{\partial W^{\text{out}}} = (y(x_n) - t_n) \cdot \frac{\partial y(x_n)}{\partial W^{\text{out}}}
\]

▶ Next: Descend to the next layer by computing

\[
\frac{\partial \text{cost}}{\partial h_3} = \frac{\partial \text{cost}}{\partial y(x_n)} \cdot \frac{\partial y(x_n)}{\partial h_3(x_n)}
\]

...and so on...
Backprop can be thought of as an engineering principle, that prescribes how to design an end-to-end trainable system from differentiable components:

- Use components which provide the methods `fprop`, `bprop` and `grad`. Then backprop can be automated.
- Well-suited for support by software frameworks.
Potential Issues

- "But what about local minima?"
- "But what about overfitting?"
- Vanishing gradients
The cost surface/local optima

- Local minima not an issue in practice
- This is probably due to high dimensional parameter space, which causes most critical points to be **saddle points** not local optima.
- Some recent theoretical work supports this view (Choromanska et al. 2014); (Dauphin, et al. 2014)

figure from wikipedia
Overfitting
Overfitting in regression

(Bishop 2006: Pattern recognition and machine learning)
Preventing overfitting in neural networks

- **Early stopping:**

![Graph showing training cost and validation cost over training iterations]

- **Weight decay** (somewhat outdated): add a weight penalty to the training objective (weight constraints now more common)

- **Dropout** (Hinton et al., 2012): Corrupt hidden unit activations during training

- **More data**

- **Weight sharing** (reduce the number of parameters):
Parameters can be shared by having them point to the same memory location.

Very common way to reduce parameters and encode prior knowledge.

Central ingredient in conv-nets (CNNs) and recurrent nets (RNNs).

Caveat: It requires long-range communication.
The vanishing gradients problem

- The backward-pass is a sequence of matrix multiplies.
- Depending on the magnitude of the eigenvalues, initial values can blow up or decay to zero.
- This can make learning difficult or slow.
- Potential solutions: architectural tricks (for example, the “LSTM” unit)
Neural nets learn distributed representations

Neural networks encode information as vectors of real values.
This makes it easy to encode conceptual similarities. In a text processing task, for example:
  - If user searches for **Dell notebook battery size**, we would like to match documents with “Dell laptop battery capacity”
  - If user searches for **Seattle motel**, we would like to match documents containing “Seattle hotel”

(Example from Chris Manning)
Summary so far

1. Non-linear pipelines are good
2. It is easy to train non-linear pipelines end-to-end using back-prop + SGD
3. Local minima are a non-issue
4. Overfitting is an issue, but it can be solved

▶ The two crucial changes that made deep learning work on real-world tasks ≈ 2010:

GPUs + Large datasets
DL impact in speech recognition

According to Microsoft’s speech group:

Using DL

Word error rate on Switchboard

1990
2000
2010

figure from Yoshua Bengio
Convolutional networks (CNN)

- LeCun et al. 1998
- The gist: Instead of feeding a large image to the network, feed small patches to the network.
- → dramatic reduction of parameters
- CNNs also have subsampling layers, so higher layers see more of the image.
ImageNet challenge 2012

**Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)**

Held in conjunction with [PASCAL Visual Object Classes Challenge 2012 (VOC2012)](http://www.pascal-network.org/challenges/VOC/)

**Back to Main page**

**All results**

- Task 1 (classification)
- Task 2 (localization)
- Task 3 (fine-grained classification)
- Team information and abstracts

**Task 1**

<table>
<thead>
<tr>
<th>Team name</th>
<th>Filename</th>
<th>Error (5 guesses)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuperVision</td>
<td>test_preds-131-137-145-135-145.txt</td>
<td>0.16422</td>
<td>Using only supplied training data</td>
</tr>
<tr>
<td>ISI</td>
<td>pred_FVs_wLACs_weighted.txt</td>
<td>0.26172</td>
<td>Weighted sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively</td>
</tr>
<tr>
<td>ISI</td>
<td>pred_FVs_weighted.txt</td>
<td>0.26602</td>
<td>Weighted sum of scores from classifiers using each FV</td>
</tr>
<tr>
<td>ISI</td>
<td>pred_FVs_summed.txt</td>
<td>0.26646</td>
<td>Naive sum of scores from classifiers using each FV</td>
</tr>
<tr>
<td>ISI</td>
<td>pred_FVs_wLACs_summed.txt</td>
<td>0.26685</td>
<td>Naive sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively</td>
</tr>
</tbody>
</table>
ImageNet challenge 2012

some first-layer features

some results

Krizhevsky, Sutskever, Hinton; 2012
High-level features

Figure 4: Top regions for six pool$_5$ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).

Girshick et al., 2014
GoogLeNet (Szegedy et al. 2014)

- Exercise in (a) scaling up, (b) unconventional architectures
- Won ImageNet 2014 with 6.66% top-5 error rate
- A variation of this network including BatchNormalization (Ioffe, Szegedy, 2015) achieves 4.8% top-5 error rate, surpassing the accuracy of human raters
### Results!

<table>
<thead>
<tr>
<th>Team Name</th>
<th>Classification accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio baseline</td>
<td>22.4 %</td>
</tr>
<tr>
<td>Video baseline</td>
<td>22.7 %</td>
</tr>
<tr>
<td>Fusion</td>
<td>27.5 %</td>
</tr>
<tr>
<td>Nottingham</td>
<td>24.7 %</td>
</tr>
<tr>
<td>Oulu</td>
<td>21.5 %</td>
</tr>
<tr>
<td>KIT</td>
<td>29.8 %</td>
</tr>
<tr>
<td>UCSD</td>
<td>37.1 %</td>
</tr>
<tr>
<td>ICT@CAS</td>
<td>35.9 %</td>
</tr>
<tr>
<td>York</td>
<td>27.6 %</td>
</tr>
<tr>
<td>LNMIIT</td>
<td>20.5 %</td>
</tr>
<tr>
<td>Montreal</td>
<td>41.0 %</td>
</tr>
<tr>
<td>Ulm</td>
<td>27.2 %</td>
</tr>
</tbody>
</table>

- **1st**: Montreal with 41.0%
- **2nd**: UCSD with 37.1%
- **3rd**: ICT@CAS with 35.9%
Conv-nets learn good *generic* features

(Donahue et al, 2013)
Word embeddings

Bengio et al 2000

This is a way to learn **distributed representations** for symbols (words).
## Word embeddings

**King - Man + Woman = Queen**

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>big - bigger</td>
<td></td>
<td></td>
<td>quick: quicker</td>
</tr>
<tr>
<td>Miami - Florida</td>
<td></td>
<td></td>
<td>Kona: Hawaii</td>
</tr>
<tr>
<td>Einstein - scientist</td>
<td></td>
<td></td>
<td>Picasso: painter</td>
</tr>
<tr>
<td>Sarkozy - France</td>
<td></td>
<td></td>
<td>Koizumi: Japan</td>
</tr>
<tr>
<td>copper - Cu</td>
<td></td>
<td></td>
<td>uranium: plutonium</td>
</tr>
<tr>
<td>Berlusconi - Silvio</td>
<td></td>
<td></td>
<td>Obama: Barack</td>
</tr>
<tr>
<td>Microsoft - Windows</td>
<td></td>
<td></td>
<td>Apple: iPhone</td>
</tr>
<tr>
<td>Microsoft - Ballmer</td>
<td></td>
<td></td>
<td>Apple: Jobs</td>
</tr>
<tr>
<td>Japan - sushi</td>
<td></td>
<td></td>
<td>USA: pizza</td>
</tr>
<tr>
<td>Germany: bratwurst</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Robotics/Reinforcement Learning

Levine et al. 2015

Mnih et al. 2013
Recurrent networks (RNN)

- Stepping the network $T$ time steps yields the equivalent of a $T$-layer feedforward net with weights that are shared between layers.
- Training the network by unrolling it in time is called back-prop-through-time (BPTT).
- Vanishing gradients especially problematic here.
Long-Short Term Memory (LSTM)

(Hochreiter, Schmidhuber; 1997)
RNN applications (thanks mainly to LSTM)

- Machine Translation (Sutskever et al. NIPS 2014), (Cho et al. Arxiv 2014)
- Speech synthesis (Fan et al. INTERSPEECH 2014)
- Speech recognition (Hannun et al., 2014)
- Handwriting generation
  http://www.cs.toronto.edu/ graves/handwriting.html
- Text generation
- Caption generation
The encoder-decoder architecture

Machine translation examples:

<table>
<thead>
<tr>
<th>Type</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our model</td>
<td>Ulrich UNK, membre du conseil d’administration du constructeur automobile Audi, affirme qu’il s’agit d’une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d’administration afin qu’ils ne soient pas utilisés comme appareils d’écoute à distance.</td>
</tr>
<tr>
<td>Truth</td>
<td>Ulrich Hackenberg, membre du conseil d’administration du constructeur automobile Audi, déclare que la collecte des téléphones portables avant les réunions du conseil, afin qu’ils ne puissent pas être utilisés comme appareils d’écoute à distance, est une pratique courante depuis des années.</td>
</tr>
<tr>
<td>Our model</td>
<td>Les téléphones cellulaires, qui sont vraiment une question, non seulement parce qu’ils pourraient potentiellement causer des interférences avec les appareils de navigation, mais nous savons, selon la FCC, qu’ils pourraient interférer avec les tours de téléphone cellulaire lorsqu’ils sont dans l’air”, dit UNK.</td>
</tr>
<tr>
<td>Truth</td>
<td>“Les téléphones portables sont véritablement un problème, non seulement parce qu’ils pourraient éventuellement créer des interférences avec les instruments de navigation, mais parce que nous savons, d’après la FCC, qu’ils pourraient perturber les antennes-relais de téléphonie mobile s’ils sont utilisés à bord”, a déclaré Rosenker.</td>
</tr>
<tr>
<td>Our model</td>
<td>Avec la crémation, il y a un “sentiment de violence contre le corps d’un être cher”, qui serait “réduit à une pile de cendres” en très peu de temps au lieu d’un processus de décomposition qui accompagnera les étapes du deuil”</td>
</tr>
<tr>
<td>Truth</td>
<td>Il y a, avec la crémation, “une violence faite au corps aimé”, qui va être “réduit à un tas de cendres” en très peu de temps, et non après un processus de décomposition, qui “accompagnerait les phases du deuil”.</td>
</tr>
</tbody>
</table>

Sutskever et al. NIPS 2014, Bahdanau et al. 2014
Caption generation (Xu et al 2015)

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.
Handwriting generation

A Symposium on High Performance Chips
A Symposium on High Performance Chips
A Symposium on High Performance Chips
A Symposium on High Performance Chips

http://www.cs.toronto.edu/~graves/handwriting.html
Naturalism and decision for the majority of Arab countries' capitalide was grounded by the Irish language by [[John Clair]], [[An Imperial Japanese Revolt]], associated with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal in the [[Protestant Immineners]], which could be said to be directly in Cantonese Communication, which followed a ceremony and set inspired prison, training. The emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known in western [[Scotland]], near Italy to the conquest of India with the conflict. Copyright was the succession of independence in the slop of Syrian influence that was a famous German movement based on a more popular servicious, non-doctrinal and sexual power post. Many governments recognize the military housing of the [[Civil Liberalization and Infantry Resolution 265 National Party in Hungary]], that is sympathetic to be to the [[Punjab Resolution]] (PJS)[http://www.humah.yahoo.com/guardian.cfm/7754800786d17551963s89.htm Official economics Adjoint for the Nazism, Montgomery was swear to advance to the resources for those Socialism's rule, was starting to signing a major tripad of aid exile.]]
Generating text

from: Andrej Karpathy:
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Roland Memisevic

Deep Learning

August 23, 2015
Lemma 0.1. Assume (3) and (5) by the construction in the description.
Suppose $X = \lim |X|$ by the formal open covering $X$ and a single map $\text{Proj}_X(A) = \text{Spec}(B)$ over $U$ compatible with the complex

$$\text{Set}(A) = \Gamma(X, \mathcal{O}_X, \mathcal{O}_X).$$

When in this case of to show that $Q \to C_{E/X}$ is stable under the following result in the second conditions of (1) and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are empty). If $T$ is surjective we may assume that $T$ is connected with residue fields of $S$. Moreover there exists a closed subspace $Z \subset X$ of $X$ where $U$ in $X'$ is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem (1) $f$ is locally of finite type. Since $S = \text{Spec}(R)$ and $Y = \text{Spec}(R)$.

Proof. This is form all sheaves of sheaves on $X$. But given a scheme $U$ and a surjective étale morphism $U \to X$. Let $U \cap U = \text{Proj} \text{Spec}(A)$ be the scheme $X$ over $S$ at the schemes $X_i \to X$ and $U = \lim_i U_i$. The following lemma surjective restrecomposes of this implies that $F_{\text{et}} = F_{\text{et}} = F_{X_{\text{et}}}$.

Lemma 0.1. Let $X$ be a locally Noetherian scheme over $S$, $E = F_{X/S}$. Set $\mathcal{L} = \mathcal{J}_{i, \mathcal{E}} / \mathcal{L}_{i, \mathcal{E}}$. Since $T$ is an $\mathcal{O}$ over $i \leq p$ is a subset of $\mathcal{L}_{i, \mathcal{E}}$. We may assume $q' = 0$.

Proof. We will use the property we see that $p$ is the next functor (??). On the other hand, by Lemma ?? we see that $D(\mathcal{O}_X) = \mathcal{O}_X(D)$ where $K$ is an $\mathcal{F}$-algebra where $\mathcal{O}_{X,D}$ is a scheme over $S$.

Proof. See discussion of sheaves of sets.

The result for prove any open covering follows from the less of Example ??, R. It may replace $S$ by $X_{\text{spcets,etale}}$, which gives an open subspace of $X$ and $T$ equal to $S_{\text{et}},$ see Descent, Lemma ??, Namely, by Lemma ?? we see that $R$ is geometrically regular over $S$.

---

from: Andrej Kaparthy:
http://karpathy.github.io/2015/05/21/rnn-effectiveness/
DRAWing (Gregor et al., 2015)
A natural generalization of a word embedding is a sentence embedding:

```
I could see the cat on the steps.
```

Query and nearest sentence:

- He ran his hand inside his coat, double-checking that the unopened letter was still there.
- He slipped his hand between his coat and his shirt, where the folded copies lay in a brown envelope.
- I'm sure you'll have a glamorous evening, she said, giving an exaggerated wink.
- I'm really glad you came to the party tonight, he said, turning to her.
- Although she could tell he hadn't been too invested in any of their other chitchat, he seemed genuinely curious about this.
- Although he hadn't been following her career with a microscope, he'd definitely taken notice of her appearances.
- An annoying buzz started to ring in my ears, becoming louder and louder as my vision began to swim.
- A weighty pressure landed on my lungs and my vision blurred at the edges, threatening my consciousness altogether.
- If he had a weapon, he could maybe take out their last imp, and then beat up Errol and Vanessa.
- If he could ram them from behind, send them sailing over the far side of the levee, he had a chance of stopping them.
- Then, with a stroke of luck, they saw the pair head together toward the portaloos.
- Then, from out back of the house, they heard a horse scream probably in answer to a pair of sharp spurs digging deep into its flanks.
- “I'll take care of it,” Goodman said, taking the phonebook.
- “I'll do that,” Julia said, coming in.
- He finished rolling up scrolls and, placing them to one side, began the more urgent task of finding ale and tankards.
- He righted the table, set the candle on a piece of broken plate, and reached for his flint, steel, and tinder.
Deep Learning as a compute paradigm

- perform a *series of operations* to solve a task
- + use *learning* to define the computations
Deep Learning as a compute paradigm

- perform a *series of operations* to solve a task
- + use *learning* to define the computations
- + make each computation *parallel*
Deep Learning as a compute paradigm

perform a series of operations to solve a task
+ use learning to define the computations
+ make each computation parallel

Dense, parallel computations are easy, if we don’t need to program them.
Deep learning needs parallel hardware.

Parallel hardware needs deep learning.
Part II: Research directions, software tools, outlook
Structured prediction

Prediction:

\[
\begin{array}{c}
\text{x} \\
\rightarrow \\
\text{y}
\end{array}
\]
Structured prediction

Prediction:

Structured prediction:

Problem: combinatorial explosion
Solution: Impose tractable dependency structure

Applications: Scene labeling, text, speech, ...
Structured prediction

Prediction:

```
x  →  y
```

Problem: combinatorial explosion

Solution: Impose tractable dependency structure

Applications: Scene labeling, text, speech, ...
Structured prediction

Prediction:

\[ x \rightarrow y \]

Problem: combinatorial explosion
Solution: Impose tractable dependency structure

\[ x \rightarrow y \]

Applications: Scene labeling, text, speech, ...
Solution proposed in 1998

- Main insight: When layers are complex *graphs*, back-prop still works (LeCun et al 1998)
- This observation was recycled in 2001 under the name *Conditional Random Field*
Streetview (Goodfellow et al, ICLR 2014)

- recent extension to recognizing text in images: eg. Jaderberg et al. ICLR 2015
Towards scene understanding

Farabet et al, 2013
Unsupervised learning
The curse of dimensionality

- There are $2^{16 \times 16}$ tiny binary images of size $16 \times 16$ pixels.
- A child of age 3 has seen less than 10 billion images and much fewer labeled images.
- How is it possible we can do any vision?
The curse of dimensionality

All natural images

All images
Unsupervised learning

Data may be distributed along some lower-dimensional \textit{manifold} in the dataspace.
Principal Components Analysis (PCA)

- If that manifold is *linear*, learning is easy and it can be done in closed form: Compute the eigenvectors of the data covariance matrix.
Autoencoders

If the manifold is non-linear (or not a really a manifold) we can use autoencoders.

Autoencoders are simple neural networks that are trained to reconstruct their input:

\[
\text{cost} = \| r(\mathbf{x}) - \mathbf{x} \|^2
\]

The hidden layer is a bottleneck that forces the model to compress the inputs.
In practice, it is more common to use overcomplete hiddens and to enforce compression in other ways (for example, by making the hidden activations sparse).
Autoencoders learn to do compression

\[ r(x) = Wh(W^T x + b) \]
Autoencoders learn to do compression

\[ r(x) = Wh(W^T x + b) \]
Stacked autoencoders (Le et al. 2012)
Other unsupervised methods

- Restricted Boltzmann machines
- Independent components analysis
- Sparse coding
- K-means clustering

Most of these models can be implemented as a form of autoencoder, or trained using their own, specialized learning criteria.
The utility of unsupervised learning

- Unsupervised learning helps when the amount of labeled data is small.
- But its utility pales in comparison to supervised back-prop on lots of data.
- Possible reasons:
  - (i) reconstruction may be the wrong objective
  - (ii) we need to scale up more
  - (iii) we need to rethink unsupervised learning
Architecture/non-linearities
M, 1994

Roland Memisevic
Deep Learning
August 23, 2015
Many tasks are based on encoding relations not things: Analogy making, motion understanding, invariance, depth estimation.

Multiplicative neurons may be a way to efficiently learn and encode such structure.
LSTM uses gating to address vanishing gradients

(Hochreiter, Schmidhuber; 1997)

- LSTM addresses the vanishing gradients problem by introducing a **constant unit** (with self-connection 1.0) surrounded by “**control logic**” (gating units).
Other RNN gating mechanisms

Cho et al 2014

Michalski et al 2014
Orthogonal transformations

\[ U^T L U = \begin{bmatrix} R_1 \\ \cdots \\ R_k \end{bmatrix} \]

\[ R_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \]

\[ x \xrightarrow{U^T} y = Lx \]
sine waves
sine waves
sine waves
sine waves
chirps

(predict-ahead interval)

(mean squared error)

(CRBM vs RNN vs grammar cells)

Roland Memisevic
Deep Learning
August 23, 2015
Form attention to differentiable models of computation and “neural programs”
Attention

Hard attention (Mnih et al, 2014)

Soft attention (Bahdanau et al, 2014)
Differentiable models of computation

- Neural Turing Machine (Graves et al, 2014)
- Memory Networks (Weston et al, 2014)
- Learning to Transduce with Unbounded Memory (Grefenstette et al. 2015)

To be able to back-propagate, all operations have to be based on differentiable operations
- (But sampling-based methods may work otherwise)
Learning to execute (Zaremba, Sutskever; 2014)

Input:

```
j=8584
    for x in range(8):
        j+=920
    b=(1500+j)
    print((b+7567))
```

Target: 25011.

Input:

```
i=8827
    c=(i-5347)
    print((c+8704) if 2641<8500 else 5308)
```

Target: 12184.
From neural networks to “neural programs”

I: Jane went to the hallway.
I: Mary walked to the bathroom.
I: Sandra went to the garden.
I: Daniel went back to the garden.
I: Sandra took the milk there.
Q: Where is the milk?
A: garden

I: Everybody is happy.
Q: What’s the sentiment?
A: positive
Q: What are the POS tags?
A: NN VBZ JJ.
I: The answer is far from obvious.
Q: In French?
A: La réponse est loin d’être évidente.

Kumar et al 2015
From neural networks to “neural programs”

Kumar et al 2015
Von Neumann via Deep Learning

- In the past we simulated neural nets on classic hardware and it didn’t work
- Today we simulate classic hardware on neural nets and it works beautifully

The benefit of today’s way: Add parallelization and your “program” may run faster and faster and faster...
Deep learning with limited precision

- Gupta et al, 2015
- Courbariaux et al, 2015:

<table>
<thead>
<tr>
<th>Format</th>
<th>Prop.</th>
<th>Up.</th>
<th>PI MNIST</th>
<th>MNIST</th>
<th>CIFAR10</th>
<th>SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single float</td>
<td>32</td>
<td>32</td>
<td>1.05%</td>
<td>0.51%</td>
<td>14.05%</td>
<td>2.71%</td>
</tr>
<tr>
<td>Half float</td>
<td>16</td>
<td>16</td>
<td>1.10%</td>
<td>0.51%</td>
<td>14.14%</td>
<td>3.02%</td>
</tr>
<tr>
<td>Fixed point</td>
<td>20</td>
<td>20</td>
<td>1.39%</td>
<td>0.57%</td>
<td>15.98%</td>
<td>2.97%</td>
</tr>
<tr>
<td>Dynamic fxp.</td>
<td>10</td>
<td>12</td>
<td>1.28%</td>
<td>0.59%</td>
<td>14.82%</td>
<td>4.95%</td>
</tr>
</tbody>
</table>
Spiking networks

- Neural network activations (real and artificial) tend to be **sparse**.
- So we are sending around, and multiplying by, lots of floating-point zeros.
- We are also applying synchronization and logic, although real brains don’t seem to.
- The logical conclusion: **spiking networks** (but it is not clear yet how to train them)

Sparsity levels in two networks trained on CIFAR-10.
(N1_Crpt, N2_Crpt trained with dropout).

figure by Kishore Konda
Back-prop using asynchronous, local computations?

In the brain, where is the backward channel?
Towards back-prop using local computations

- Hinton 2007: Use the temporal derivative to encode the error derivative!
- (see also: Bengio et al. 2015)
- Recall that the derivative of most common cost functions is, conveniently, given by

\[
\frac{\partial \text{cost}}{\partial y(x_n)} = y(x_n) - t_n
\]

How local back-prop may work

- Let top-layer drive the activations towards the correct value.
- Let feedback weights transport that change downward.
- Make weight changes proportional to the rate of change of a postsynaptic neuron and the value of the pre-synaptic neuron.
Is the brain doing local back-prop?

- (Hinton 2007): “What would neuro-scientists see if this is what’s happening in the brain?”
- They should see this (and they do!):

![Diagram](http://www.scholarpedia.org/article/Spike-timing_dependent_plasticity)

Roland Memisevic  
Deep Learning  
August 23, 2015
Research directions

- Applications
- Architectures (attention, vanishing gradients, neural programs)
- Reinforcement learning
- Theory
- Scaling up, hardware
- Multimodality and grounding: vision, language, speech, robotics
Tricks and facts

▶ Do not be afraid of non-differentiabilities (or even discontinuities). They don’t matter.
▶ Gradient clipping (constraining the norm of the gradients) helps avoid getting thrown out by NaNs too often (especially for recurrent nets).
▶ Batch-normalization (Ioffe, Szegedy; 2015) helps training: Normalize hidden unit activations to have fixed means/standard deviations during training, by drawing the statistics from the current mini-batch.
▶ Adding a “momentum-term” to your SGD updates can have a very strong influence on convergence speed.
▶ You rarely successfully “try out” a model on a new task, you make the model work on the task.
Deep Learning Software/Frameworks

- Back-prop: torch, theano
- Add-ons: blocks, fuel, lasagne
- Low-level: cuDNN, nervanagpu, cudamat
- Convnets: caffe, overfeat, cuda-convnet, sklearn-theano
- Word embeddings: word2vec (available in gensim)
```python
print(__doc__)
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Rectangle
from sklearn.datasets import load_sample_image
from sklearn.feature_extraction import OverFeatClassifier

X = load_sample_image("sloth_closeup.jpg")
top_n_classes = 5
clf = OverFeatClassifier(top_n=top_n_classes)
prediction = clf.predict(X)

# Shortened the labels for plotting
prediction = [p.split(",") for p in prediction.ravel()]

f, axarr = plt.subplots(2, 1)
plt.suptitle("Top %i classification and cropping box" % top_n_classes)
axarr[0].imshow(X)
axarr[0].add_patch(crop_box)
axarr[0].autoscale(enable=False)
axarr[0].get_xaxis().set_ticks([])
axarr[0].get_yaxis().set_ticks([])
ind = np.arange(top_n_classes)
width = .45
axarr[1].bar(ind, prediction_probs.ravel(), width, color='steelblue')
axarr[1].set_xticks(ind + width / 2)
axarr[1].set_xticklabels(prediction, rotation='vertical')
axarr[1].set_ylabel("Probability")
plt.show()
```
Welcome

Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently. Theano features:

- **tight integration with NumPy** – Use numpy.ndarray arrays in Theano-compiled functions.
- **transparent use of a GPU** – Perform data-intensive calculations up to 140x faster than with CPU (float32 only).
- **efficient symbolic differentiation** – Theano does your derivatives for function with one or many inputs.
- **speed and stability optimizations** – Get the right answer for log(1+x) even when x is really tiny.
- **dynamic C code generation** – Evaluate expressions faster.
- **extensive unit-testing and self-validation** – Detect and diagnose many types of mistake.

Theano has been powering large-scale computationally intensive scientific investigations since 2007. But it is also approachable enough to be used in the classroom (IFT6266 at the University of Montreal).

News

- Open Machine Learning Workshop 2014 presentation.
- Colin Raffel tutorial on Theano.
- Ian Goodfellow did a 12h class with exercises on Theano.
- Theano 0.6 was released. Everybody is encouraged to update.
- New technical report on Theano: Theano: new features and speed improvements. However, please keep citing the other paper below in scientific work involving Theano.
- HPCC 2011 Tutorial. We included a few fixes discovered while doing the Tutorial.

You can watch a quick (20 minute) introduction to Theano given as a talk at SciPy 2010 via streaming (or downloaded) video.


http://www.deeplearning.net/tutorial/
import theano
from theano import tensor

a = tensor.dscalar()
b = tensor.dscalar()

c = a + b

f = theano.function([a,b], c)

assert 4.0 == f(1.5, 2.5)
import theano
import theano.tensor as T

#define computational graph:
W = T.dmatrix()
inputs = T.dmatrix()
targets = T.dmatrix()
outputs = T.dot(W.T, inputs)
cost_theano = ((outputs - targets)**2).mean()
grad_theano = T.grad(cost_theano, W)

#compile functions:
cost = theano.function([W, inputs, targets], cost_theano)
grad = theano.function([W, inputs, targets], grad_theano)

#try on some _random_ data:
my_w = 0.01*randn(10,1).astype("float32")
my_inputs = randn(100,10).T.astype("float32")
my_targets = randn(1,100).astype("float32")
print cost(my_w, my_inputs, my_targets)
my_w -= 0.1*grad(my_w, my_inputs, my_targets)
print cost(my_w, my_inputs, my_targets)
prehiddens = T.dot(inputs, W)
hiddens = (prehiddens > selectionthreshold) * prehiddens
outputs = T.dot(hiddens, W.T) + bvis

cost = T.mean(T.sum(0.5 * ((inputs - outputs)**2), axis=1))
grad = T.grad(cost, params)
Thank you

Questions?

Deep learning needs parallel hardware.

Parallel hardware needs deep learning.