Deep Learning, Recurrent Nets and Attention for System 2 Processing

Yoshua Bengio

July 19th, 2018

ACDL’2018 – Lecture 2
Anything New with Deep Learning since the Neural Nets of the 90s?

• Rectified linear units instead of sigmoids, enable training much deeper networks by backprop (Glorot & Bengio AISTATS 2011)

• Some forms of noise (like dropout) are powerful regularizers yielding superior generalization abilities

• Success of deep convnets trained on large labeled image datasets, success of skip connections (ResNets)

• Success of recurrent nets with more memory, with gating units

• Success of word embedding, neural machine translation, deep NLP

• Attention mechanisms liberate neural nets from fixed-size inputs

• Autoencoders, adversarial training, generating images & sounds

• Transfer learning, meta-learning, deep reinforcement learning
Google Image Search: Different object types represented in the same space

Google:
S. Bengio, J. Weston & N. Usunier

Learn $\Phi_1(\cdot)$ and $\Phi_w(\cdot)$ to optimize precision@k.
Maps Between Representations

$x$ and $y$ represent different modalities, e.g., image, text, sound...

Can provide 0-shot generalization to new categories (values of $y$)

(Larochelle et al AAAI 2008)
Multi-Task Learning

- Generalizing better to new tasks (tens of thousands!) is crucial to approach AI

- Deep architectures learn good intermediate representations that can be shared across tasks
  
  (Collobert & Weston ICML 2008, Bengio et al AISTATS 2011)

- Good representations that disentangle underlying factors of variation make sense for many tasks because each task concerns a subset of the factors

Prior: shared underlying explanatory factors between tasks
Combining Multiple Sources of Evidence with Shared Representations

- Traditional ML: data = matrix
- Relational learning: multiple sources, different tuples of variables
- Share representations of same types across data sources
- Shared learned representations help propagate information among data sources: e.g., WordNet, XWN, Wikipedia, FreeBase, ImageNet...
  (Bordes et al AISTATS 2012, ML J. 2013)
- FACTS = DATA
- Deduction = Generalization
Recurrent Neural Networks

- Can produce an output at each time step: unfolding the graph tells us how to back-prop through time.

\[
\begin{align*}
O &= V s + W x \\
V &= U x \\
W &= V \\
S_t &= W S_{t-1} + V O_{t-1} \\
O_t &= W S_t + V O_t \\
S_{t+1} &= W S_t + V O_t \\
O_{t+1} &= W S_{t+1} + V O_{t+1}
\end{align*}
\]
Recurrent Neural Networks

- Selectively summarize an input sequence in a fixed-size state vector via a recursive update

\[ s_t = F_\theta(s_{t-1}, x_t) \]

\[ s_t = G_t(x_t, x_{t-1}, x_{t-2}, \ldots, x_2, x_1) \]

- Generalizes naturally to new lengths not seen during training
Generative RNNs

• An RNN can represent a fully-connected **directed generative model**: every variable predicted from all previous ones.

\[
P(x) = P(x_1, \ldots x_T) = \prod_{t=1}^{T} P(x_t|x_{t-1}, x_{t-2}, \ldots x_1)
\]

\[
L_t = -\log P(x_t|x_{t-1}, x_{t-2}, \ldots x_1)
\]
Conditional Distributions

- Sequence to vector
- Sequence to sequence of the same length, aligned
- Vector to sequence
- Sequence to sequence
During training, past $y$ in input is from training data

At generation time, past $y$ in input is generated

Mismatch can cause "compounding error"

$\hat{y}_t \sim P(y_t \mid h_t)$

$(x_t, y_t)$: next input/output training pair
Ideas to reduce the train/generate mismatch in teacher forcing

• Scheduled sampling (*S. Bengio et al, NIPS 2015*)

• Backprop through open-loop sampling recurrence & minimize long-term cost (but which one? GAN would be most natural → Professor Forcing)

Related to SEARN (Daumé et al 2009) DAGGER (Ross et al 2010)

Gradually increase the probability of using the model’s samples vs the ground truth as input.
Increasing the Expressive Power of RNNs with more Depth

- ICLR 2014, *How to construct deep recurrent neural networks*

Ordinary RNNs

+ stacking

+ deep hid-to-out
+ deep hid-to-hid
+ deep in-to-hid

+ skip connections for creating shorter paths
Bidirectional RNNs, Recursive Nets, Multidimensional RNNs, etc.

The unfolded architecture needs not be a straight chain

Recursive (tree-structured) Neural Nets:
- Frasconi et al 97
- Socher et al 2011

Bidirectional RNNs (Schuster and Paliwal, 1997)

See Alex Graves’s work, e.g., 2012

(Multidimensional RNNs, Graves et al 2007)
**Multiplicative Interactions**


- Multiplicative Integration RNNs:
  - Replace
    
    \[ \phi(Wx + Uz + b) \]
  - By
    
    \[ \phi(Wx \odot Uz + b) \]
  - Or more general:
    
    \[ \phi(\alpha \odot Wx \odot Uz + \beta_1 \odot Uz + \beta_2 \odot Wx + b) \]
Multiscale or Hierarchical RNNs

(Bengio & Elhihi, NIPS 1995)

• Motivation :
  o Gradients can propagate over longer spans through slow time-scale paths

• Approach :
  o Introduce a network architecture that update the states of its hidden layers with different speeds in order to capture multiscale representation of sequences.
Learning Long-Term Dependencies with Gradient Descent is Difficult

How to store 1 bit? Dynamics with multiple basins of attraction in some dimensions

- Some subspace of the state can store 1 or more bits of information if the dynamical system has multiple basins of attraction in some dimensions.

Note: gradients MUST be high near the boundary.
Robustly storing 1 bit in the presence of bounded noise

- With spectral radius \(>1\), noise can kick state out of attractor

  \[ \text{UNSTABLE} \]

  Domain of \(\alpha_t\)

- Not so with radius \(<1\)

  \[ \text{CONTRACTIVE} \rightarrow \text{STABLE} \]

  Domain of \(\alpha_t\)
Storing Reliably ➔ Vanishing gradients

- Reliably storing bits of information requires spectral radius < 1
- The product of $T$ matrices whose spectral radius is < 1 is a matrix whose spectral radius converges to 0 at exponential rate in $T$

\[
L = L(s_T(s_{T-1}(\ldots s_{t+1}(s_t, \ldots))))
\]

\[
\frac{\partial L}{\partial s_t} = \frac{\partial L}{\partial s_T} \frac{\partial s_T}{\partial s_{T-1}} \ldots \frac{\partial s_{t+1}}{\partial s_t}
\]

- If spectral radius of Jacobian is < 1 ➔ propagated gradients vanish
Vanishing or Exploding Gradients

- Hochreiter’s 1991 MSc thesis (in German) had independently discovered that backpropagated gradients in RNNs tend to either vanish or explode as sequence length increases.
Why it hurts gradient-based learning

- Long-term dependencies get a weight that is exponentially smaller (in $T$) compared to short-term dependencies

\[
\frac{\partial C_t}{\partial W} = \sum_{\tau \leq t} \frac{\partial C_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W} = \sum_{\tau \leq t} \frac{\partial C_t}{\partial a_t} \frac{\partial a_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W}
\]

Becomes exponentially smaller for longer time differences, when spectral radius < 1
Vanishing Gradients in Deep Nets are Different from the Case in RNNs

- If it was just a case of vanishing gradients in deep nets, we could just rescale the per-layer learning rate, but that does not really fix the training difficulties.

- Can’t do that with RNNs because the weights are shared, & total true gradient = sum over different “depths”

\[
\frac{\partial C_t}{\partial W} = \sum_{\tau \leq t} \frac{\partial C_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W} = \sum_{\tau \leq t} \frac{\partial C_t}{\partial a_t} \frac{\partial a_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W}
\]
To store information robustly the dynamics must be contractive

- The RNN gradient is a product of Jacobian matrices, each associated with a step in the forward computation. To store information robustly in a finite-dimensional state, the dynamics must be contractive [Bengio et al 1994].

\[
L = L(s_T(s_{T-1}(\ldots s_{t+1}(s_t, \ldots))))
\]

\[
\frac{\partial L}{\partial s_t} = \frac{\partial L}{\partial s_T} \frac{\partial s_T}{\partial s_{T-1}} \ldots \frac{\partial s_{t+1}}{\partial s_t}
\]

- Problems:
  - e-values of Jacobians > 1 \(\rightarrow\) gradients explode
  - or e-values < 1 \(\rightarrow\) gradients shrink & vanish
  - or random \(\rightarrow\) variance grows exponentially

Storing bits robustly requires e-values < 1

Gradient clipping
Dealing with Gradient Explosion by Gradient Norm Clipping

\[ \hat{g} \leftarrow \frac{\partial \text{error}}{\partial \theta} \]

\[ \text{if } \| \hat{g} \| \geq \text{threshold} \text{ then } \]

\[ \hat{g} \leftarrow \frac{\text{threshold}}{\| \hat{g} \|} \hat{g} \]

\text{end if}

(Mikolov thesis 2012; Pascanu, Mikolov, Bengio, ICML 2013)
RNN Tricks
(Pascanu, Mikolov, Bengio, ICML 2013; Bengio, Boulanger & Pascanu, ICASSP 2013)

• Clipping gradients (avoid exploding gradients)
• Leaky integration (propagate long-term dependencies)
• Momentum (cheap 2\textsuperscript{nd} order)
• Initialization (start in right ballpark avoids exploding/vanishing)
• Sparse Gradients (symmetry breaking)
• Gradient propagation regularizer (avoid vanishing gradient)
• Gated self-loops (LSTM & GRU, reduces vanishing gradient)
Delays & Hierarchies to Reach Farther

- Delays and multiple time scales, *Elhihi & Bengio NIPS 1995, Koutnik et al ICML 2014*
- How to do this right?
- How to automatically and adaptively do it?

Hierarchical RNNs (words / sentences): *Sordoni et al CIKM 2015, Serban et al AAAI 2016*
Fighting the vanishing gradient: LSTM & GRU

(Hochreiter 1991); first version of the LSTM, called Neural Long-Term Storage with self-loop

- Create a path where gradients can flow for longer with a self-loop
- Corresponds to an eigenvalue of Jacobian slightly less than 1
- LSTM is now heavily used (Hochreiter & Schmidhuber 1997)
- GRU light-weight version (Cho et al 2014)
Gating for Attention-Based Neural Machine Translation

Related to earlier Graves 2013 for generating handwriting

- (Bahdanau, Cho & Bengio, arXiv sept. 2014)
- (Jean, Cho, Memisevic & Bengio, arXiv dec. 2014)

\[ f = (\text{La, croissance, économique, s'est, ralentie, ces, dernières, années, .}) \]

\[ e = (\text{Economic, growth, has, slowed, down, in, recent, years, .}) \]
What’s New with Deep Learning?

- Incorporating the idea of **attention, using GATING units**, has unlocked a breakthrough in machine translation:

  Neural Machine Translation  (ICLR’2015)

- Now in Google Translate:

  - Softmax over lower locations conditioned on context at lower and higher locations

  - Human evaluation
Graph Attention Networks
Velickovic et al, ICLR 2018

- Handle variable-size neighborhood of each node using the same neural net by using an attention mechanism to aggregate information from the neighbors
- Use multiple attention heads to collect different kinds of information

Figure 1: Left: The attention mechanism \( a(W \tilde{h}_i, W \tilde{h}_j) \) employed by our model, parametrized by a weight vector \( \tilde{a} \in \mathbb{R}^{2F_0} \), applying a LeakyReLU activation.

Right: An illustration of multi-head attention (with \( K = 3 \) heads) by node 1 on its neighborhood. Different arrow styles and colors denote independent attention computations. The aggregated features from each head are concatenated or averaged to obtain \( \tilde{h}_0 \).

To stabilize the learning process of self-attention, we have found extending our mechanism to employ multi-head attention to be beneficial, similarly to Vaswani et al. (2017). Specifically, \( K \)-independent attention mechanisms execute the transformation of Equation 4, and then their features are concatenated, resulting in the following output feature representation:

\[
\tilde{h}_0 = \bigoplus_{k=1}^{K} \tilde{h} = \bigoplus_{k=1}^{K} \left( \tilde{h}_0 \bigoplus_{j \in N(i)} \sum_{i} \tilde{a}_{ij} W_k \tilde{h}_j \right)
\]

where \( \bigoplus \) represents concatenation, \( \tilde{a}_{ij} \) are normalized attention coefficients computed by the \( k \)-th attention mechanism \( (a_k) \), and \( W_k \) is the corresponding input linear transformation’s weight matrix.

Note that, in this setting, the final returned output, \( h_0 \), will consist of \( KF_0 \) features (rather than \( F_0 \)) for each node.

Specially, if we perform multi-head attention on the final (prediction) layer of the network, concatenation is no longer sensible—instead, we employ averaging, and delay applying the final nonlinearity (usually a softmax or logistic sigmoid for classification problems) until then:

\[
\tilde{h}_0 = \sum_{k=1}^{K} \tilde{a}_{ij} W_k \tilde{h}_j
\]
What’s New with Deep Learning?

• Attention has also opened the door to neural nets which can write to and read from a memory
  • 2 systems:
    • Cortex-like (state controller and representations)
      • System 1, intuition, fast heuristic answer
    • Hippocampus-like (memory) + prefrontal cortex
      • System 2, slow, logical, sequential
  
• Memory-augmented networks gave rise to
  • Systems which reason
    • Sequentially combining several selected pieces of information (from the memory) in order to obtain a conclusion
  
• Systems which answer questions
  • Accessing relevant facts and combining them
Attention Mechanisms for Memory Access

- Neural Turing Machines *(Graves et al 2014)*
- and Memory Networks *(Weston et al 2014)*
- Use a content-based attention mechanism *(Bahdanau et al 2014)* to control the read and write access into a memory
- The attention mechanism outputs a softmax over memory locations

\[
\alpha = \frac{e^{f_i(h)}}{\sum_i e^{f_i(h)}}
\]

\[
\mathbf{r} = \sum_i \alpha_i \mathbf{c}_i
\]

Read = weighted average of attended contents
Large Memory Networks: Sparse Access Memory for Long-Term Dependencies

- Memory = part of the state
- Memory-based networks are special RNNs
- A mental state stored in an external memory can stay for arbitrarily long durations, until it is overwritten (partially or not)
- Forgetting = vanishing gradient.
- Memory = **higher-dimensional state**, avoiding or reducing the need for forgetting/vanishing
The next word generated can either come from vocabulary or is copied from the input sequence.

Table 5: Europarl Dataset (EN-FR)

<table>
<thead>
<tr>
<th>Machine Translation</th>
<th>BLEU-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMT</td>
<td>20.19</td>
</tr>
<tr>
<td>NMT + PS</td>
<td><strong>23.76</strong></td>
</tr>
</tbody>
</table>

Table 3: Results on Gigaword Corpus for modeling UNK’s with pointers in terms of recall.

<table>
<thead>
<tr>
<th></th>
<th>Rouge-1</th>
<th>Rouge-2</th>
<th>Rouge-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMT + lvt</td>
<td>36.45</td>
<td>17.41</td>
<td>33.90</td>
</tr>
<tr>
<td>NMT + lvt + PS</td>
<td><strong>37.29</strong></td>
<td><strong>17.75</strong></td>
<td><strong>34.70</strong></td>
</tr>
</tbody>
</table>
Variational Hierarchical RNNs for Dialogue Generation (Serban et al 2016)

- Lower level = words of an utterance (turn of speech)
- Upper level = state of the dialogue
- Inject high-level choices

---

Figure 1: 'omputational graph for V%áK3 model. Bounded boxes represent deterministic real-valued vectors. Variables $z$ represent latent stochastic variables. Vt training time for $n = 1, ..., N$ a sample $z_n$ is drawn from the approximate posterior $N(\mu_{\text{posterior}}(w_1, ..., w_n), \sigma_{\text{posterior}}(w_1, ..., w_n))$ and used to estimate the gradient of the variational lower-bound given by $K_q$. The approximate posterior is parametrized by its own one-layer feedforward neural network, which takes as input the output of the context at the current time step as well as the output of the encoder for the next sub-sequence.

The V%áK3 model greatly helps to reduce the problems with the generation process used by the 'JJ?' and %áK3 model outlined above. The variation of the output sequence is now modelled in two ways: at the sequence-level with the conditional prior distribution over $z$ and at the sub-sequence level (token-level) with the conditional distribution over tokens $w_1, ..., w_M$. The variable $z$ helps model long-term output trajectories by representing high-level information about the sequence, which in turn allows the variable $h$ to primarily focus on summarizing the information up to token $M$. Intuitively, the randomness injected by the variable $z$ corresponds to higher-level decisions like topic or sentiment of the sentence.

---

Experimental Evaluation

We consider the problem of conditional natural language response generation for dialogue. This is an interesting problem with applications in areas such as customer service, technical support, language learning and entertainment. It is also a task domain that requires learning to generate sequences with complex structures while taking into account long-term context.

We consider two tasks. For each task, the model is given a dialogue context, consisting of one or more utterances, and the goal of the model is to generate an appropriate next response to the dialogue.

We first perform experiments on a Twitter Dialogue Corpus. The task is to generate utterances to append to existing Twitter conversations. The dataset is extracted using a procedure similar toitter et al. and is split into training, validation, and test sets, containing respectively 749, 060, 936, 633, and 10,000 dialogues. Each dialogue contains 6.27 utterances and 94.16 tokens on average. The dialogues are fairly long compared to recent large-scale language modelling corpora, such as the —illion Word 'model —enchmark, which focus on modelling single sentences.

We also experiment on the Ubuntu Dialogue Corpus, which contains about 500,000 dialogues extracted from the Ubuntu !nternet áelayed 'hat channel. Users enter the chat channel with a Ubuntu-related technical problem, and other users try to help them. For further details see Appendix M5j5. We chose these corpora because they are large, and have different purposes—Ubuntu dialogues are typically goal driven, whereas Twitter dialogues typically contain social interaction.
Still Far from Human-Level AI

• Industrial successes mostly based on supervised learning

• Learning superficial clues, not generalizing well outside of training contexts, easy to fool trained networks:
  – Current models cheat by picking on surface regularities

• Still unable to discover higher-level abstractions
How to Discover Good Disentangled Representations

• How to discover abstractions?
• What is a good representation? *(Bengio et al 2013)*
• Need clues (= priors) to help disentangle the underlying factors, such as
  • Spatial & temporal scales
  • Marginal independence
  • Simple dependencies between factors
    • Consciousness prior
  • Causal / mechanism independence
    • Controllable factors
Acting to Guide Representation Learning & Disentangling

(E. Bengio et al, 2017; V. Thomas et al, 2017)

- Some factors (e.g. objects) correspond to ‘independently controllable’ aspects of the world

- Can only be discovered by acting in the world

  - Control linked to notion of objects & agents

  - Causal but agent-specific & subjective: affordances
Abstraction Challenge for Unsupervised Learning

• Why is modeling \( P(\text{acoustics}) \) so much worse than modeling \( P(\text{acoustics} \mid \text{phonemes}) P(\text{phonemes}) \)?

• Wrong level of abstraction?
  - many more entropy bits in acoustic details then linguistic content

  → predict the future in abstract space instead: non-trivial
Conscious thoughts are very low-dimensional objects compared to the full state of the (unconscious) brain.

Yet they have unexpected predictive value or usefulness.

→ strong constraint or prior on the underlying representation.

**Thought**: composition of few selected factors / concepts (key/value) at the highest level of abstraction of our brain.

Richer than but closely associated with short verbal expression such as a **sentence** or phrase, a **rule** or **fact** (link to classical symbolic AI & knowledge representation).
How to select a few relevant abstract concepts making a thought?

Content-based Attention
On the Relation between Abstraction and Attention

- Attention allows to focus on a few elements out of a large set
- Soft-attention allows this process to be trainable with gradient-based optimization and backprop

Attention focuses on a few appropriate abstract or concrete elements of mental representation
The Consciousness Prior
Bengio 2017, arXiv:1709.08568

- 2 levels of representation:
  - High-dimensional abstract representation space (all known concepts and factors) $h$
  - Low-dimensional conscious thought $c$, extracted from $h$

- $c$ includes names (keys) and values of fac...
Disentangling up to Linear Projection

• My old view of disentangling: each dimension of the representation = one ‘nameable’ (semantic) factor

• Potential problem: the number of ‘nameable’ factors is limited by the number of units, and brains don’t use a completely localized representation for named things

• My current view of disentangling: it is enough that a linear projection exist to ‘classify’ or ‘predict’ any of the factors

• The ‘number’ of potential ‘nameable’ factors is now exponentially larger (e.g. subsets of dimensions, weights of these projections)
The Consciousness Prior
Bengio 2017, arXiv:1709.08568

- Conscious prediction over attended variables $A$ (soft attention)

$$V = - \sum_{A} w_A \log P(h_{t,A} = a | c_{t-1})$$

- Attention weights
- Factor name
- Predicted value
- Earlier conscious state

Conscious state $c$

Unconscious state $h$

Input $x$
What Training Objective?

- How to train the attention mechanism which selects which variables to predict?
  - Representation learning without reconstruction:
    - Maximize entropy of code
    - Maximize mutual information between past and future
  - **Objective function completely in abstract space, higher-level parameters model dependencies in abstract space**
  - **Usefulness of thoughts: as conditioning information for action, i.e., a particular form of planning for RL, i.e., the estimated gradient of rewards could also be used to drive learning of abstract representations**