Artificial Neural Networks

II

Ronan Collobert
ronan@collobert.com
Summary

McCulloch and Pitts
- Boolean functions
- No training

Perceptron
- Linear classification
- Convergence if separable
- Generalization?

AdaLine
- Linear classification/regression
- Delta Rule
- Convergence?

Kernel Perceptron
- Non-linear classification

SVM
- Linear classification
- Non-linear with kernels
- Margin: better generalization?

Multi Layer Perceptron
- Non-linear classification/regression
- Gradient descent (backprop)
  - Convergence?
  - Generalization?

Unsupervised Training
- Reconstruction bottleneck:
  - layer size
  - sparsity
  - transpose constraint

Specializations
- RBF
- Convolutions 1D/2D
- Sequence classification

Margin Perceptron
- Linear classification
- Margin: better generalization?
Part I

Generalization
Generalization

Polynomial d=1

From (Bottou, 2010)
Polynomial $d=3$

From (Bottou, 2010)
Polynomial d=20

From (Bottou, 2010)
Generalization: VC dim

- Bound the difference train-test error given “complexity” measure of class of functions

- $h$ is the Vapnik-Chervonenkis dimension
- $L$ training examples
- With probability $1 - \eta$:

$$\text{testerr} \leq \text{trainerr} + \sqrt{\frac{h(\log(2L/h) + 1) - \log(\eta/4)}{L}}$$

$$(1974)$$
**VC dim** of a set of functions: maximum number of points $L$ that can be separated into two different classes in all the $2^L$ ways

From (Burges, 1998)

- **VC dim** \{ linear classifiers $x \mapsto w \cdot x$, dim $d$ \}: $h = d + 1$
- **VC dim** \{ linear classifiers with margin $\geq \rho$, dim $d$ \}: $h \leq \min\left(\frac{R^2}{\rho^2}, d\right) + 1$
- **VC dim** \{ neural net classifiers with $n$ parameters \}: $h \sim O(n^4)$
  (Karpinski & Macintyre, 1997)
Gradient Descent Convergence
Proofs from (Bottou, 1991)

Given a cost function $C(w)$, we perform

$$w^{t+1} = w^t - \lambda^t \frac{\partial C(w^t)}{\partial w}$$

Assume we have a single minimum $w^*$ and

$$\forall \epsilon \quad \inf_{||w - w^*||^2 > \epsilon} (w - w^*) \frac{\partial C(w)}{\partial w} > 0$$

Define sequence

$$h^t = (w^t - w^*)^2$$

Idea: if $u_t \geq 0$ and $\sum_t (u_{t+1} - u_t)_+ < \infty$ then $u_t$ converges

Consider

$$h^{t+1} - h^t = -2 \lambda^t (w^t - w^*) \frac{\partial C(w^t)}{\partial w} + \left( \lambda^t \frac{\partial C(w^t)}{\partial w} \right)^2$$
Consider

\[
h^{t+1} - h^t = -2\lambda^t(w^t - w^*) \frac{\partial C(w^t)}{\partial w} + \left(\lambda^t \frac{\partial C(w^t)}{\partial w}\right)^2
\]

Assume

\[
\left(\frac{\partial C(w)}{\partial w}\right)^2 \leq A + B(w - w^*)^2 \quad (A, B \geq 0)
\]

Then we get:

\[
h^{t+1} - h^t \leq A(\lambda^t)^2 + B(\lambda^t)^2 h^t \quad \Rightarrow \quad h^{t+1} - (1 + B(\lambda^t)^2) h^t \leq A(\lambda^t)^2
\]

Assume

\[
\sum_t (\lambda^t)^2 < \infty
\]

The following sequence converges:

\[
\mu^t = \prod_{i=1}^{t} \frac{1}{1 + B(\lambda^i)^2}
\]

We have \(\mu^t h^{t+1} - \mu^{t-1} h^t \leq A(\lambda^t)^2 \mu^t\)

\* So \(\sum_t A(\lambda^t)^2 \mu^t < \infty\)

\* \(\Rightarrow \mu^{t-1} h^t\) converges

\* \(\Rightarrow h^t\) converges
We have

$$h^{t+1} - h^t = -2\lambda^t (w^t - w^*) \frac{\partial C(w^t)}{\partial w} + \left( \lambda^t \frac{\partial C(w^t)}{\partial w} \right)^2$$

$h^t$ converges and $\sum_t (\lambda^t)^2 < \infty$, so with previous assumption

$$\sum_t \lambda^t (w^t - w^*) \frac{\partial C(w^t)}{\partial w} < \infty$$

Make sure learning rates do not decrease too quickly:

$$\sum_t \lambda^t = \infty$$

In that case $(w^t - w^*) \frac{\partial C(w^t)}{\partial w}$ converges to 0, and because of initial assumption

$$w^t \to w^*$$
(Stochastic) Gradient Descent Convergence \(1/2\)

- Given a cost function \(C(w)\), we perform
  \[
  w^{t+1} = w^t - \lambda^t H(z^t, w^t)
  \]
  such that
  \[
  \mathbb{E}_z H(z, w^t) = \frac{\partial C(w^t)}{\partial w}
  \]

- **Same idea** than before, with **same kind of hypothesis**, but this time
  \[
  h^t = (w^t - w^*)^2
  \]
  is a **random variable**.

- **Use the same kind of “trick”:**
  if \(u_t \geq 0\) and \(\sum_t \mathbb{E}(\delta_t (u_{t+1} - u_t)) < \infty\) then \(u_t\) converges a.s.
  with
  \[
  \delta_t = \begin{cases} 
  1 & \text{if } \mathbb{E}(u^{t+1} - u^t | \mathcal{P}^t) > 0 \\
  0 & \text{otherwise}
  \end{cases}
  \]

  where \(\mathcal{P}^t\) is the “history” up to time \(t\)
  \[
  \mathcal{P}^t = z^0, \ldots, z^{t-1}, w^0, \ldots, w^t, \lambda^0, \ldots, \lambda^t
  \]
More general convergence theorems exist (Bottou, 1991)

★ Assume $C(w)$ is three time differentiable

★ If several minima, then we can show $w^t$ stay “confined” in the same region when $\lambda^t$ decreases.

★ Assume $C \geq C_{\min}$ and consider $h^t = C(w^t) - C_{\min}$

Assumptions similar than before:

$$\sum_t \lambda^t = \infty \quad \text{and} \quad \sum_t (\lambda^t)^2 < \infty$$

and

$$\mathbb{E}_z(H(z, w))^2 \leq A + B w^2 \quad \text{with} \quad A, B \geq 0$$

Then we get

$$C(w^t) \to C^\infty \quad \text{a.s.} \quad \text{and} \quad \left(\frac{\partial C(w^t)}{\partial w}\right)^2 \to 0 \quad \text{a.s.}$$
Applications
Audio: Continuous Speech Recognition

Fig. 1. baseline TDNN

Fig. 3. 2 Word MS-TDNN

Fig. 4. Phone Model for ‘p’

From (Haffner, 1992)
Specialized Training: Non-Linear CRF

- **Sequence** of \( T \) frames \([x]_1^T\)
- The network score for class \( k \) at the \( t^{th} \) frame is \( f([x]_1^T, k, t, \theta)\)
- \( A_{kl} \) transition score to jump from class \( k \) to class \( l \)

\[
\begin{align*}
    s([x]_1^T, [i]_1^T, \tilde{\theta}) &= \sum_{t=1}^{T} \left( A_{[i]_{t-1}[i]_t} + f([x]_1^T, [i]_t, t, \theta) \right) \\
    \log p([y]_1^T | [x]_1^T, \tilde{\theta}) &= s([x]_1^T, [y]_1^T, \tilde{\theta}) - \log \text{add} \ s([x]_1^T, [j]_1^T, \tilde{\theta}) \quad \forall [j]_1^T
\end{align*}
\]
Normalization computed with recursive **Forward** algorithm:

\[ \delta_t(j) = \text{logAdd}_i \left[ \delta_{t-1}(i) + A_{i,j} + f_{\theta}(j, x_1^T, t) \right] \]

Termination:

\[ \text{logadd \ } s([x]_1^T, [j]_1^T, \tilde{\theta}) = \text{logAdd}_i \delta_T(i) \]

- Simply backpropagate through this recursion with chain rule

**Non-linear CRFs:** **Graph Transformer Networks** (Bottou et al., 1997)

- Compared to CRFs, we train features (network parameters \( \theta \) and transitions scores \( A_{kl} \))

- Inference: **Viterbi** algorithm (replace \( \text{logAdd} \) by \( \text{max} \))
Fig 10.2 - Mise en commun des parties identiques. Aux transitions en gras sont associés les mots “pavement” et “parement”.

From (Bottou, 1991)
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

<table>
<thead>
<tr>
<th>Method</th>
<th>Err. rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian SVM</td>
<td>1.4</td>
</tr>
<tr>
<td>1000 HU NN (MSE)</td>
<td>4.5</td>
</tr>
<tr>
<td>800 HU NN</td>
<td>1.6</td>
</tr>
<tr>
<td>CNN</td>
<td>0.8</td>
</tr>
<tr>
<td>CNN + distortions</td>
<td>0.4</td>
</tr>
<tr>
<td>6 layers NN + distortions</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Fig. 4. Size-normalized examples from the MNIST database.
Image: Digit Recognition

(Lecun et al., 1998)
Viterbi Penalty

\( G_{vit} \)

\( T_{vit} \)

Viterbi Transformer

Interpretation Graph

\( G_{int} \)

\( T_{rec} \)

Recognition Transformer

\( G_{seg} \)

Segmentation Graph

(Lecun et al., 1998)
Image: Face Detection

(C) Osadchy et al., 2007
Image: Face Detection
Image: Object Recognition

![Image of object recognition examples](image-url)

<table>
<thead>
<tr>
<th>exp#</th>
<th>Classifier</th>
<th>Input</th>
<th>Dataset</th>
<th>Test Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Linear</td>
<td>raw 2x96x96</td>
<td>norm-unif</td>
<td>30.2%</td>
</tr>
<tr>
<td>1.1</td>
<td>K-NN (K=1)</td>
<td>raw 2x96x96</td>
<td>norm-unif</td>
<td>18.4%</td>
</tr>
<tr>
<td>1.2</td>
<td>K-NN (K=1)</td>
<td>PCA 95</td>
<td>norm-unif</td>
<td>16.6%</td>
</tr>
<tr>
<td>1.3</td>
<td>SVM Gauss</td>
<td>raw 2x96x96</td>
<td>norm-unif</td>
<td>N.C.</td>
</tr>
<tr>
<td>1.4</td>
<td>SVM Gauss</td>
<td>raw 1x48x48</td>
<td>norm-unif</td>
<td>13.9%</td>
</tr>
<tr>
<td>1.5</td>
<td>SVM Gauss</td>
<td>raw 1x32x32</td>
<td>norm-unif</td>
<td>12.6%</td>
</tr>
<tr>
<td>1.6</td>
<td>SVM Gauss</td>
<td>PCA 95</td>
<td>norm-unif</td>
<td>13.3%</td>
</tr>
<tr>
<td>1.7</td>
<td>Conv Net 80</td>
<td>raw 2x96x96</td>
<td>norm-unif</td>
<td>6.6%</td>
</tr>
<tr>
<td>1.8</td>
<td>Conv Net 100</td>
<td>raw 2x96x96</td>
<td>norm-unif</td>
<td>6.8%</td>
</tr>
<tr>
<td>2.0</td>
<td>Linear</td>
<td>raw 2x96x96</td>
<td>jitt-unif</td>
<td>30.6%</td>
</tr>
<tr>
<td>2.1</td>
<td>Conv Net 100</td>
<td>raw 2x96x96</td>
<td>jitt-unif</td>
<td>7.1%</td>
</tr>
</tbody>
</table>

Detection/Segmentation/Recognition

<table>
<thead>
<tr>
<th>exp#</th>
<th>Classifier</th>
<th>Input</th>
<th>Dataset</th>
<th>Test Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Conv Net 100</td>
<td>raw 2x96x96</td>
<td>jitt-text</td>
<td>10.6%</td>
</tr>
<tr>
<td>6.0</td>
<td>Conv Net 100</td>
<td>raw 2x96x96</td>
<td>jitt-clutt</td>
<td>16.7%</td>
</tr>
<tr>
<td>6.2</td>
<td>Conv Net 100</td>
<td>raw 1x96x96</td>
<td>jitt-clutt</td>
<td>39.9%</td>
</tr>
</tbody>
</table>

(LeCun et al., 2004)
Text: Natural Language Processing (Tasks)

- Part-Of-Speech Tagging (POS): syntactic roles (noun, adverb...)
- Chunking (CHK): syntactic constituents (noun phrase, verb phrase...)
- Name Entity Recognition (NER): person/company/location...
- Semantic Role Labeling (SRL): semantic role

\[ \text{[John]}_{ARG0} \ [\text{ate}]_{REL} \ [\text{the apple}]_{ARG1} \ [\text{in the garden}]_{ARGM-LOC} \]

- Parsing (PSG):

- Tagging tasks (BIOES tagging scheme):

  The black cat sat on the mat .
  B-NP I-NP E-NP S-VP S-PP B-NP E-NP O
### Standard NLP Benchmarks

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS (Toutanova, 2003)</td>
<td>Various combinations of surrounding words &amp; tags, various caps, digit, dash, various prefixes &amp; suffixes. <strong>Dependency Network</strong></td>
</tr>
<tr>
<td>Chunking (Sha, 2003)</td>
<td>Surrounding words, POS tags. <strong>Conditional Random Field (CRF)</strong></td>
</tr>
<tr>
<td>NER (Ando, 2005)</td>
<td>Surrounding words, POS, several suffixes &amp; prefixes, surrounding tags, bigrams, previously assigned tags to words, unlabeled data. <strong>Viterbi decoding at test</strong></td>
</tr>
<tr>
<td>SRL (Koomen, 2005)</td>
<td>6 parse trees, pruning heuristics, POS, voice, phrase type, head words, subparts of the trees, ... <strong>Argument identification, argument classification, integer linear programming</strong></td>
</tr>
</tbody>
</table>
Lexicalized Probabilistic Context-Free Grammar (PCFG), POS, head words, chart parser, deleted interpolation, ... 30 pages of details in (Bikel, 2004)

Re-ranking over the above, using lots of ad-hoc features

PCFG, dependency features

CRF or similar
Words into Vectors

A word = index in a dictionary

The cat sat on the mat = \((w_1, w_2, w_3, w_4, w_5, w_6)\)

**binary code \(\sim\) dictionary size**

\[
w \leftrightarrow \begin{pmatrix}
0, \cdots, 0, & 1, & 0, \cdots, 0
\end{pmatrix}^T = (\mathbf{1}_{=w})^T
\]

**word embedding**

\[M \sim \text{feature size} \times \text{dictionary size}\]

\[M \times (\mathbf{1}_{=w}) = M \bullet w\]

lookup-table operation

**sentence embedding**

\[M \times (\mathbf{1}_{=w_1} \cdots \mathbf{1}_{=w_6}) = (M \bullet w_1 \cdots M \bullet w_6)\]

**Convolution** (kernel size 1)

Applicable to any discrete feature (words, caps, stems...)

See (Bengio et al, 2001)
How to tag “in” in the sentence “The Visigoths settled in southern Gaul”?

Window Approach
How to tag “in” in the sentence “The Visigoths settled in southern Gaul”?

Window Approach (extra features)
How to tag “in” in the sentence “The Visigoths settled in southern Gaul”?
Max Over Time

For each $i$, what is the chosen $t$?

$$\max_t [X]_{i,t} \quad \forall i$$
Ranking Language Model

- **Language Model:** “is a sentence actually english or not?”
  Implicitly captures: ⋆ syntax ⋆ semantics

- Bengio & Ducharme (2001) Probability of next word given previous words. Overcomplicated – we do not need probabilities here

- Entropy criterion largely determined by most frequent phrases

- Rare legal phrases are no less significant that common phrases

- $f()$ a window approach network

- **Ranking** margin cost:

\[
\sum_{s \in S} \sum_{w \in D} \max(0, 1 - f(s, w^*_s) + f(s, w))
\]

  $S$: sentence windows  
  $D$: dictionary  
  $w^*_s$: true middle word in $s$  
  $f(s, w)$: network score for sentence $s$ and middle word $w$

- **Stochastic** training:
  ⋆ positive example: random corpus sentence  
  ⋆ negative example: replace middle word by random word
Training Language Model

- Two window approach (11) networks (100HU) trained on two corpus:
  - LM1: Wikipedia: **631M** of words
  - LM2: Wikipedia + Reuters RCV1: **631M + 221M = 852M** of words

- Massive dataset: cannot afford classical training-validation scheme

- Like in biology: breed a couple of network lines

- Breeding decisions according to 1M words validation set

- LM1
  - order dictionary words by frequency
  - increase dictionary size: 5000, 10,000, 30,000, 50,000, 100,000
  - 4 weeks of training

- LM2
  - initialized with LM1, dictionary size is **130,000**
  - 30,000 additional most frequent Reuters words
  - 3 additional weeks of training
<table>
<thead>
<tr>
<th>country</th>
<th>word</th>
<th>word</th>
<th>word</th>
<th>word</th>
<th>word</th>
<th>word</th>
</tr>
</thead>
<tbody>
<tr>
<td>france</td>
<td>jesus</td>
<td>xbox</td>
<td>reddish</td>
<td>scratched</td>
<td>megabits</td>
<td></td>
</tr>
<tr>
<td>454</td>
<td>1973</td>
<td>6909</td>
<td>11724</td>
<td>29869</td>
<td>87025</td>
<td></td>
</tr>
<tr>
<td>austria</td>
<td>god</td>
<td>amiga</td>
<td>greenish</td>
<td>nailed</td>
<td>octets</td>
<td></td>
</tr>
<tr>
<td>belgium</td>
<td>sati</td>
<td>playstation</td>
<td>bluish</td>
<td>smashed</td>
<td>mb/s</td>
<td></td>
</tr>
<tr>
<td>germany</td>
<td>christ</td>
<td>msx</td>
<td>pinkish</td>
<td>punched</td>
<td>bit/s</td>
<td></td>
</tr>
<tr>
<td>italy</td>
<td>satan</td>
<td>ipod</td>
<td>purplish</td>
<td>popped</td>
<td>baud</td>
<td></td>
</tr>
<tr>
<td>greece</td>
<td>kali</td>
<td>sega</td>
<td>brownish</td>
<td>crimped</td>
<td>carats</td>
<td></td>
</tr>
<tr>
<td>sweden</td>
<td>indra</td>
<td>psNUMBER</td>
<td>greyish</td>
<td>scraped</td>
<td>kbit/s</td>
<td></td>
</tr>
<tr>
<td>norway</td>
<td>vishnu</td>
<td>hd</td>
<td>grayish</td>
<td>screwed</td>
<td>megahertz</td>
<td></td>
</tr>
<tr>
<td>europe</td>
<td>ananda</td>
<td>dreamcast</td>
<td>whitish</td>
<td>sectioned</td>
<td>megapixels</td>
<td></td>
</tr>
<tr>
<td>hungary</td>
<td>parvati</td>
<td>geforce</td>
<td>silver</td>
<td>slashed</td>
<td>gbit/s</td>
<td></td>
</tr>
<tr>
<td>switzerland</td>
<td>grace</td>
<td>capcom</td>
<td>yellowish</td>
<td>ripped</td>
<td>amperes</td>
<td></td>
</tr>
</tbody>
</table>
Semi-Supervised Benchmark Results

- Initialize word embeddings with LM1 or LM2
- Same training procedure

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Benchmark Systems</strong></td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+WLL</td>
<td>96.31</td>
<td>89.13</td>
<td>79.53</td>
<td>54.53</td>
</tr>
<tr>
<td>NN+SLL</td>
<td>96.37</td>
<td>90.33</td>
<td>81.47</td>
<td>71.24</td>
</tr>
<tr>
<td>NN+WLL+LM1</td>
<td>97.05</td>
<td>91.91</td>
<td>85.68</td>
<td>57.32</td>
</tr>
<tr>
<td>NN+SLL+LM1</td>
<td>97.10</td>
<td>93.65</td>
<td>87.58</td>
<td>74.28</td>
</tr>
<tr>
<td>NN+WLL+LM2</td>
<td>97.14</td>
<td>92.04</td>
<td>86.96</td>
<td>56.97</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>73.90</td>
</tr>
</tbody>
</table>

- Huge boost from language models
- Training set word coverage:

<table>
<thead>
<tr>
<th></th>
<th><strong>LM1</strong></th>
<th><strong>LM2</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>POS</td>
<td>97.86%</td>
<td>98.83%</td>
</tr>
<tr>
<td>CHK</td>
<td>97.93%</td>
<td>98.91%</td>
</tr>
<tr>
<td>NER</td>
<td>95.50%</td>
<td>98.95%</td>
</tr>
<tr>
<td>SRL</td>
<td>97.98%</td>
<td>98.87%</td>
</tr>
</tbody>
</table>
Multi-Task Learning

- Joint training
- Good overview in (Caruana, 1997)
# Multi-Task Learning Benchmark Results

## Window Approach

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHK (F1)</th>
<th>NER (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
</tr>
<tr>
<td>NN+SLL+LM2+MTL</td>
<td>97.22</td>
<td>94.10</td>
<td>88.62</td>
</tr>
</tbody>
</table>

## Sentence Approach

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.12</td>
<td>93.37</td>
<td>88.78</td>
<td>73.90</td>
</tr>
<tr>
<td>NN+SLL+LM2+MTL</td>
<td>97.22</td>
<td>93.72</td>
<td>87.99</td>
<td>74.33</td>
</tr>
</tbody>
</table>
Cascading Tasks

Increase level of engineering by incorporating common NLP techniques

- **Stemming** for western languages benefits **POS** (Ratnaparkhi, 1996)
  - Use *last two characters* as feature (455 different stems)

- **Gazetteers** are often used for **NER** (Florian, 2003)
  - 8,000 locations, person names, organizations and misc entries from CoNLL 2003

- **POS** is a good feature for **CHK & NER** (Shen, 2005) (Florian, 2003)
  - We feed our *own POS* tags as feature

- **CHK** is also a common feature for **SRL** (Koomen, 2005)
  - We feed our *own CHK* tags as feature
<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>73.90</td>
</tr>
<tr>
<td>NN+SLL+LM2+Suffix2</td>
<td>97.29</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+Gazetteer</td>
<td>–</td>
<td>–</td>
<td>89.59</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+POS</td>
<td>–</td>
<td>94.32</td>
<td>88.67</td>
<td>75.39</td>
</tr>
<tr>
<td>NN+SLL+LM2+CHK</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>74.73</td>
</tr>
</tbody>
</table>
Parsing is essential to SRL (Punyakanok, 2005) (Pradhan, 2005)

- State-of-the-art SRL systems use several parse trees (up to 6!!)
- We feed our network several levels of Charniak parse tree provided by CoNLL 2005
<table>
<thead>
<tr>
<th>Approach</th>
<th>SRL (test set F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Benchmark System</strong> (six parse trees)</td>
<td>77.92</td>
</tr>
<tr>
<td><strong>Benchmark System</strong> (top Charniak only)</td>
<td>74.76†</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>73.90</td>
</tr>
<tr>
<td>NN+SLL+LM2+CHK</td>
<td>74.73</td>
</tr>
<tr>
<td>NN+SLL+LM2+Charniak (level 1 only)</td>
<td>76.27</td>
</tr>
<tr>
<td>NN+SLL+LM2+Charniak (levels 1 &amp; 2)</td>
<td>76.24</td>
</tr>
<tr>
<td>NN+SLL+LM2+Charniak (levels 1 to 3)</td>
<td>76.62</td>
</tr>
<tr>
<td>NN+SLL+LM2+Charniak (levels 1 to 4)</td>
<td>76.50</td>
</tr>
<tr>
<td>NN+SLL+LM2+Charniak (levels 1 to 5)</td>
<td>76.98</td>
</tr>
</tbody>
</table>

† on the validation set