CSE 573: Artificial Intelligence

Hanna Hajishirzi
Neural Networks

slides adapted from
Dan Klein, Pieter Abbeel ai.berkeley.edu
And Dan Weld, Luke Zettlemoyer
Reminder: Linear Classifiers

- Inputs are feature values
- Each feature has a weight
- Sum is the activation

\[ \text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x) \]

- If the activation is:
  - Positive, output +1
  - Negative, output -1
How to get probabilistic decisions?

- **Activation:** $z = w \cdot f(x)$
- **If** $z = w \cdot f(x)$ very positive $\rightarrow$ want probability going to 1
- **If** $z = w \cdot f(x)$ very negative $\rightarrow$ want probability going to 0
- **Sigmoid function**

$$
\phi(z) = \frac{1}{1 + e^{-z}}
$$
Best w?

- Maximum likelihood estimation:

\[
\max_w \quad ll(w) = \max_w \sum_i \log P(y^{(i)} | x^{(i)}; w)
\]

with:

\[
P(y^{(i)} = +1 | x^{(i)}; w) = \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}
\]

\[
P(y^{(i)} = -1 | x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}
\]

= Logistic Regression
**Multiclass Logistic Regression**

- **Multi-class linear classification**
  - A weight vector for each class: \( w_y \)
  - Score (activation) of a class \( y \): \( w_y \cdot f(x) \)
  - Prediction w/highest score wins: \( y = \arg \max_y w_y \cdot f(x) \)

- How to make the scores into probabilities?

\[
\begin{align*}
    z_1, z_2, z_3 &\rightarrow \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}}, \quad \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}}, \quad \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}
\end{align*}
\]

original activations \hspace{1cm} softmax activations
Best $w$?

- Maximum likelihood estimation:

$$\max_w \ ll(w) = \max_w \sum_i \log P(y^{(i)} | x^{(i)}; w)$$

with:

$$P(y^{(i)} | x^{(i)}; w) = \frac{e^{w_y^{(i)} \cdot f(x^{(i)})}}{\sum_y e^{w_y \cdot f(x^{(i)})}}$$

= Multi-Class Logistic Regression
This Lecture

- Optimization

- i.e., how do we solve:

\[
\max_w \ ll(w) = \max_w \sum_i \log P(y^{(i)} | x^{(i)}; w)
\]
Hill Climbing

- simple, general idea
  - Start wherever
  - Repeat: move to the best neighboring state
  - If no neighbors better than current, quit

- What’s particularly tricky when hill-climbing for multiclass logistic regression?
  - Optimization over a continuous space
    - Infinitely many neighbors!
  - How to do this efficiently?
Mini-Batch Gradient Ascent on the Log Likelihood Objective

\[
\max_w \ ll(w) = \max_w \sum_i \log P(y^{(i)}|x^{(i)}; w)
\]

**Observation:** gradient over small set of training examples (=mini-batch) can be computed in parallel, might as well do that instead of a single one

- **init** \( w \)
- **for** iter = 1, 2, ...
  - pick random subset of training examples \( J \)
  \[
  w \leftarrow w + \alpha \sum_{j \in J} \nabla \log P(y^{(j)}|x^{(j)}; w)
  \]
How about computing all the derivatives?

- We’ll talk about that once we covered neural networks, which are a generalization of logistic regression.
Neural Networks
Multi-class Logistic Regression

- special case of neural network

\[ P(y_1|x; w) = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
\[ P(y_2|x; w) = \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
\[ P(y_3|x; w) = \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
Deep Neural Network = Also learn the features!

\[ P(y_1|x; w) = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
\[ P(y_2|x; w) = \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
\[ P(y_3|x; w) = \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}} \]
Deep Neural Network = Also learn the features!

\[ f_1(x) \]
\[ f_2(x) \]
\[ f_3(x) \]
\[ \vdots \]
\[ f_K(x) \]

\[ x_1 \]
\[ x_2 \]
\[ x_3 \]
\[ \vdots \]
\[ x_L \]

\[ z_1^{(1)} \]
\[ z_2^{(1)} \]
\[ z_3^{(1)} \]
\[ \vdots \]
\[ z_L^{(1)} \]

\[ z_1^{(2)} \]
\[ z_2^{(2)} \]
\[ z_3^{(2)} \]
\[ \vdots \]
\[ z_L^{(2)} \]

\[ z_1^{(3)} \]
\[ z_2^{(3)} \]
\[ z_3^{(3)} \]
\[ \vdots \]
\[ z_L^{(3)} \]

\[ z_1^{(n-1)} \]
\[ z_2^{(n-1)} \]
\[ z_3^{(n-1)} \]
\[ \vdots \]
\[ z_L^{(n-1)} \]

\[ z_1^{(n)} \]
\[ z_2^{(n)} \]
\[ z_3^{(n)} \]
\[ \vdots \]
\[ z_L^{(n)} \]

\[ z_1^{\text{OUT}} \]
\[ z_2^{\text{OUT}} \]
\[ z_3^{\text{OUT}} \]
\[ \vdots \]
\[ z_L^{\text{OUT}} \]

\[ g = \text{nonlinear activation function} \]

\[ z_i^{(k)} = g \left( \sum_j W_{i,j}^{(k-1,k)} z_j^{(k-1)} \right) \]
Deep Neural Network = Also learn the features!

\[
Z_i^{(k)} = g\left(\sum_j W_{i,j}^{(k-1,k)} Z_j^{(k-1)}\right)
\]

\( g = \) nonlinear activation function
Common Activation Functions

**Sigmoid Function**

\[ g(z) = \frac{1}{1 + e^{-z}} \]

\[ g'(z) = g(z)(1 - g(z)) \]

**Hyperbolic Tangent**

\[ g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \]

\[ g'(z) = 1 - g(z)^2 \]

**Rectified Linear Unit (ReLU)**

\[ g(z) = \max(0, z) \]

\[ g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases} \]
Training the deep neural network is just like logistic regression:

\[
\max_w \ ll(w) = \max_w \sum_i \log P(y^{(i)} | x^{(i)} ; w)
\]

$\hat{w}$ tends to be a much, much larger vector 😊

→just run gradient ascent

+ stop when log likelihood of hold-out data starts to decrease
Theorem (Universal Function Approximators). A two-layer neural network with a sufficient number of neurons can approximate any continuous function to any desired accuracy.

Practical considerations

- Can be seen as learning the features
- Large number of neurons
  - Danger for overfitting
  - (hence early stopping!)
Fun Neural Net Demo Site

- Demo-site:
  - http://playground.tensorflow.org/
How about computing all the derivatives?

- Derivatives tables:

  \[
  \frac{d}{dx}(a) = 0 \\
  \frac{d}{dx}(x) = 1 \\
  \frac{d}{dx}(au) = a \frac{du}{dx} \\
  \frac{d}{dx}(u + v - w) = \frac{du}{dx} + \frac{dv}{dx} - \frac{dw}{dx} \\
  \frac{d}{dx}(uv) = u \frac{dv}{dx} + v \frac{du}{dx} \\
  \frac{d}{dx}\left(\frac{u}{v}\right) = \frac{\frac{du}{dx}v - u \frac{dv}{dx}}{v^2} \\
  \frac{d}{dx}(u^n) = nu^{n-1} \frac{du}{dx} \\
  \frac{d}{dx}(\sqrt{u}) = \frac{1}{2\sqrt{u}} \frac{du}{dx} \\
  \frac{d}{dx}(\frac{1}{u}) = -\frac{1}{u^2} \frac{du}{dx} \\
  \frac{d}{dx}\left(\frac{1}{u^n}\right) = -\frac{n}{u^{n+1}} \frac{du}{dx} \\
  \frac{d}{dx}\left[\begin{array}{c} f(u) \\ u^n \end{array}\right] = \frac{d}{du}\left[\begin{array}{c} f(u) \\ u^n \end{array}\right] \frac{du}{dx} \\
  \frac{d}{dx}\ln u = \frac{1}{u} \frac{du}{dx} \\
  \frac{d}{dx}\log_a u = \frac{1}{u \ln a} \frac{du}{dx} \\
  \frac{d}{dx}e^u = e^u \frac{du}{dx} \\
  \frac{d}{dx}a^u = a^u \ln a \frac{du}{dx} \\
  \frac{d}{dx}\left(\frac{u}{v}\right) = \frac{\frac{du}{dx}v - u \frac{dv}{dx}}{v^2} + \ln u \frac{u^2}{2u} \frac{dv}{dx} \\
  \frac{d}{dx}\sin u = \cos u \frac{du}{dx} \\
  \frac{d}{dx}\cos u = -\sin u \frac{du}{dx} \\
  \frac{d}{dx}\tan u = \sec^2 u \frac{du}{dx} \\
  \frac{d}{dx}\cot u = -\csc^2 u \frac{du}{dx} \\
  \frac{d}{dx}\sec u = \sec u \tan u \frac{du}{dx} \\
  \frac{d}{dx}\csc u = -\csc u \cot u \frac{du}{dx}
  \]

[source: http://hyperphysics.phy-astr.gsu.edu/hbase/Math/derfunc.html]
How about computing all the derivatives?

- But neural net $f$ is never one of those?
  - No problem: CHAIN RULE:

$$f(x) = g(h(x))$$

Then

$$f'(x) = g'(h(x))h'(x)$$

→ Derivatives can be computed by following well-defined procedures
Automatic Differentiation

- **Automatic differentiation software**
  - e.g. Theano, TensorFlow, PyTorch, Chainer
  - Only need to program the function $g(x,y,w)$
  - Can automatically compute all derivatives w.r.t. all entries in $w$

- Need to know this exists
- How this is done? -- outside of scope of CSE573
Summary of Key Ideas

- Optimize probability of label given input
  \[ \max_w ll(w) = \max_w \sum_i \log P(y^{(i)}|x^{(i)}; w) \]

- Continuous optimization
  - Gradient ascent:
    - Compute steepest uphill direction = gradient (= just vector of partial derivatives)
    - Take step in the gradient direction
    - Repeat (until held-out data accuracy starts to drop = “early stopping”)

- Deep neural nets
  - Last layer = still logistic regression
  - Now also many more layers before this last layer
    - = computing the features
    - \( \rightarrow \) the features are learned rather than hand-designed
  - Universal function approximation theorem
    - If neural net is large enough
    - Then neural net can represent any continuous mapping from input to output with arbitrary accuracy
    - But remember: need to avoid overfitting / memorizing the training data \( \rightarrow \) early stopping!
  - Automatic differentiation gives the derivatives efficiently (how? = outside of scope of 573)
How well does it work?
How well does it work?
Next: More Neural Net Applications!
Computer Vision
Manual Feature Design
Features and Generalization

[HoG: Dalal and Triggs, 2005]
Features and Generalization

Image

HoG
Performance

ImageNet Error Rate 2010-2014

graph credit Matt Zeiler, Clarifai
Performance

ImageNet Error Rate 2010-2014

Graph credit: Matt Zeiler, Clarifai
Performance

ImageNet Error Rate 2010-2014

[Graph showing error rate from 2010 to 2014 with different error rates for Traditional CV and Deep Learning. AlexNet is highlighted in 2012 with a low error rate.]
Performance

ImageNet Error Rate 2010-2014

- Traditional CV
- Deep Learning

AlexNet

graph credit Matt Zeiler, Clarifai
Performance

ImageNet Error Rate 2010-2014

- Traditional CV
- Deep Learning

Error Rate

- 79%
- 60%
- 40%
- 20%
- 7%

Year

- 2010
- 2011
- 2012
- 2013
- 2014

AlexNet

Graph credit: Matt Zeiler, Clarifai
"man in black shirt is playing guitar."
"construction worker in orange safety vest is working on road."
"two young girls are playing with lego toy."
"boy is doing backflip on wakeboard."
"girl in pink dress is jumping in air."
"black and white dog jumps over bar."
"young girl in pink shirt is swinging on swing."
"man in blue wetsuit is surfing on wave."

Karpathy & Fei-Fei, 2015; Donahue et al., 2015; Xu et al, 2015; many more
Visual QA Challenge
Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh
Image Segmentation
Super Bowl 48 was an American football game to determine the champion of the National Football League (NFL) for the 2013 season. The National Football Conference champions Seattle Seahawks defeated the American Football Conference champions Denver Broncos. The Seahawks defeated the Broncos 43—8, the largest margin victory for an underdog and tied the third largest point differential overall (35) in Super Bowl history with Super Bowl XXVII (1993). It was the first time the winning scored over 40 points, while holding their opponent to under 10.

Question
Which NFL team represented the NFC at Super Bowl 48?

Answer
Seattle Seahawks
Machine Translation

Google Neural Machine Translation (in production)

Encoder:

```
| e0 | e1 | e2 | e3 | e4 | e5 | e6 |
```

Decoder:

```
| d0 | d1 | d2 | d3 |
```

Knowledge is power
Pipeline Approach for Question Answering

- Feature engineering
- Classifying phrases

Super Bowl 48 was an American football game to determine the champion of the National Football League (NFL) for the 2013 season. The National Football Conference champions Seattle Seahawks defeated the American Football Conference champions Denver Broncos. The Seahawks defeated the Broncos 43—8, the largest margin victory for an underdog and tied the third largest point differential overall (35) in Super Bowl history with Super Bowl XXVII (1993). It was the first time the winning scored over 40 points, while holding their opponent to under 10.

Which NFL team represented the NFC at Super Bowl 48?

words, types, frequencies dependency relations

\[ f_1, f_2, \ldots, f_n \]
Pipeline Approach Results

- Dataset: Stanford Question Answering Dataset (SQuAD) [Rajpurkar et al 2016]:
  - 100k Wikipedia documents with question
- Accuracy: percentage of correctly predicted phrases
Neural Approach

Find a function that assigns a high score to the correct answer given the context and question.

Super Bowl 48 was an American football game to determine the champion of the National Football League (NFL) for the 2013 season. The National Football Conference champions Seattle Seahawks defeated the American Football Conference champions Denver Broncos. The Seahawks defeated the Broncos 43—8, the largest margin victory for an underdog and tied the third largest point differential overall (35) in Super Bowl history with Super Bowl XXVII (1993). It was the first time the winning scored over 40 points, while holding their opponent to under 10.

Which NFL team represented the NFC at Super Bowl 48?
The National Football Conference champions Seattle Seahawks defeated the American Football Conference champions Denver Broncos. Super Bowl 48 was an American football game to determine the champion of the National Football League (NFL) for the 2013 season. The National Football Conference champions Seattle Seahawks defeated the American Football Conference champions Denver Broncos. The Seahawks defeated the Broncos 43—8, the largest margin victory for an underdog and tied the third largest point differential overall (35) in Super Bowl history with Super Bowl XXVII (1993). It was the first time the winning scored over 40 points, while holding their opponent to under 10.

Which NFL team represented the NFC at Super Bowl 48?

Seattle Seahawks
## Question Answering Leaderboard

### Jan 1, 2017

Since the release of our dataset (and paper), the community has made rapid progress! Here are the ExactMatch (EM) and F1 scores of the best models evaluated on the test and development sets of v1.1.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Test EM</th>
<th>Test F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BIDAF (ensemble)</td>
<td>73.3</td>
<td>81.1</td>
</tr>
<tr>
<td></td>
<td>Allen Institute for AI &amp; University of Washington&lt;br&gt;(Seo et al. '16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Dynamic Coattention Networks (ensemble)</td>
<td>71.6</td>
<td>80.4</td>
</tr>
<tr>
<td></td>
<td>Salesforce Research&lt;br&gt;(Xiong &amp; Zhong et al. '16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>r-net (ensemble)</td>
<td>72.1</td>
<td>79.7</td>
</tr>
<tr>
<td></td>
<td>Microsoft Research Asia</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>r-net (single model)</td>
<td>68.4</td>
<td>77.5</td>
</tr>
<tr>
<td></td>
<td>Microsoft Research Asia</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>BIDAF (single model)</td>
<td>68.0</td>
<td>77.3</td>
</tr>
<tr>
<td></td>
<td>Allen Institute for AI &amp; University of Washington&lt;br&gt;(Seo et al. '16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Multi-Perspective Matching (ensemble)</td>
<td>68.2</td>
<td>77.2</td>
</tr>
<tr>
<td></td>
<td>IBM Research</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### March 8, 2021

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FPNet (ensemble)</td>
<td>90.871</td>
<td>93.183</td>
</tr>
<tr>
<td></td>
<td>Ant Service Intelligence Team</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>IE-Net (ensemble)</td>
<td>90.758</td>
<td>93.044</td>
</tr>
<tr>
<td></td>
<td>RICOH_SRCB_DML</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SA-Net on Albert (ensemble)</td>
<td>90.724</td>
<td>93.011</td>
</tr>
<tr>
<td></td>
<td>QIANXIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>SA-Net-V2 (ensemble)</td>
<td>90.679</td>
<td>92.948</td>
</tr>
<tr>
<td></td>
<td>QIANXIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Retro-Reader (ensemble)</td>
<td>90.578</td>
<td>92.978</td>
</tr>
<tr>
<td>4</td>
<td>FPNet (ensemble)</td>
<td>90.600</td>
<td>92.899</td>
</tr>
<tr>
<td></td>
<td>YuLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>EntitySpanFocusV2 (ensemble)</td>
<td>90.521</td>
<td>92.824</td>
</tr>
<tr>
<td></td>
<td>RICOH_SRCB_DML</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ATRLP+PV (ensemble)</td>
<td>90.442</td>
<td>92.877</td>
</tr>
<tr>
<td></td>
<td>Hithink RoyalFlush</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ELECTRA+ALBERT+EntitySpanFocus (ensemble)</td>
<td>90.442</td>
<td>92.839</td>
</tr>
<tr>
<td></td>
<td>SRCB_DML</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Recurrent Neurons to encode and decode (i.e. generate) sequential data like TEXT.

An unrolled recurrent neural network.

Long Short-Term Memory (LSTM) cell
Attention helps resolve the **Vanishing Gradient Problem** that recurrent neural networks suffer over LONG SEQUENCES. At any time step, the model can decide which tokens to pay attention to from other time-steps.
Attention is all you need!

Turns out you only need attention, and can get rid of the recurrent neurons entirely!

**Transformers** : Interleaving attention layers and fully-connected layers, which can be computed **parallelly** over the sequence, instead of recurrently.

**Positional Embeddings** : Encode Sequence Information
Pretrain-then-finetune paradigm

**Pre-train** transformer on **Masked Language Modeling.**

**Finetune** transformer on task-specific data and labels.

Randomly masked: A quick [MASK] fox jumps over the [MASK] dog

Predict: A quick brown fox jumps over the lazy dog
BERT and Family

Different Variations on Transformer architectures and different pre-training tasks
Language **Understanding** tasks
- GLUE/SuperGLUE Benchmarks
- Natural Language Entailment, Paraphrase detection, Sentiment/review classification
- Question Answering, Reading Comprehension

Language **Generating** tasks
- Machine Translation
- Long-text summarization

**Dialogue Systems**: Interactive systems that have to **understand** humans and **generate** responses
Pretrained Models (BERT) on GLUE Benchmarks

Human Performance on GLUE
Massive Pre-trained models are few-shot learners! (GPT-3)

175B GPT-3 can work without fine-tuning, when it is shown sample **demonstrations** for a task:

**Few-shot**

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

1. Translate English to French:  
   2. sea otter => loutre de mer  
   3. peppermint => menthe poivrée  
   4. plush giraffe => girafe peluche  
   5. cheese => 

---

**Figure 1: Exponential growth of number of parameters in DL models**
We develop an algorithm that can detect pneumonia from chest X-rays at a level exceeding practicing radiologists.

Chest X-rays are currently the best available method for diagnosing pneumonia, playing a crucial role in clinical care and epidemiological studies. Pneumonia is responsible for more than 1 million hospitalizations and 50,000 deaths per year in the US alone.
Google and DeepMind are using AI to predict the energy output of wind farms

To help make that energy more valuable to the power grid

By Nick Statt | @nickstatt | Feb 26, 2019, 2:42pm EST

Google announced today that it has made energy produced by wind farms more viable using the artificial intelligence software of its London-based subsidiary DeepMind. By using DeepMind’s machine learning algorithms to predict the wind output from the farms Google uses for its green energy initiatives, the company says it can now schedule set deliveries of energy output, which are more valuable to the grid than standard, non-time-based deliveries.
Change in Programming Paradigm!

**Traditional Programming:** program by writing lines of code

```
If Number = 1 Then
  Count1 = Count1 + 1
Else
  If Number = 2 Then
    Count2 = Count2 + 1
  Else
    If Number = 3 then
      Count3 = Count3 + 1
    Else
      CountX = CountX + 1
    End If
  End If
End If
```

Poor performance on AI problems

**Deep Learning ("Software 2.0"):** program by providing data

Success!