Backpropagation

Linear-time DP for derivatives:
1. Write composite fn as labeled acyclic hypergraph
2. Forward propagation with input
3. Backprop: $\frac{\partial y_i}{\partial x_j} = \sum_{p \in P(j,i)} \prod_{(k \to \ell) \in p} \frac{\partial z_\ell}{\partial z_k}$

$\sin'(x) = \cos(x)$, $\cos'(x) = -\sin(x)$, $\log'(x) = \frac{1}{x}$, $\exp'(x) = \exp(x)$

Log-linear Modelling

$\mathrm{score}(y, x) = \boldsymbol{\theta}^\top \boldsymbol{f}(x, y)$
NLL gradient = 0: $\sum_{i=1}^{n} \boldsymbol{f}(x_i, y_i) = \sum_{i=1}^{n} \mathbb{E}_{y|x_i,\boldsymbol{\theta}}[\boldsymbol{f}(x_i, y)]$
Hessian: $\boldsymbol{H}_{\boldsymbol{\theta}}\left(\sum_i -\log p(y_i|x_i)\right) = \sum_i \mathrm{Cov}_{y|x_i,\boldsymbol{\theta}}[\boldsymbol{f}(x_i, y)]$
Softmax: $\mathrm{softmax}(\boldsymbol{h})_y = \frac{\exp(h_y/T)}{\sum_{y'} \exp(h_{y'}/T)}$ $T \to 0$: argmax. $T \to \infty$: uniform.
Exponential family: $p(x|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta}^\top \boldsymbol{\varphi}(x))$

Multi-layer Perceptron

Problem: Data must be linearly separable. **Solution**: Learn non-linear feature fn with MLP:
$\boldsymbol{h}_k = \sigma_k(\boldsymbol{W}_k^\top \boldsymbol{h}_{k-1})$, $\boldsymbol{h}_1 = \sigma_1(\boldsymbol{W}_1^\top \boldsymbol{e}(x))$
Then $\mathrm{softmax}(\boldsymbol{\theta}^\top \boldsymbol{h}_n)$ for prob dist.
Skip-Gram: predict if 2 words in same context. Need good word repr.
Derivative: $\frac{\partial \ell}{\partial \boldsymbol{W}_k} = \frac{\partial \ell}{\partial y} \frac{\partial y}{\partial \boldsymbol{h}_n} \left(\prod_{m=k+1}^{n} \sigma'_m(\ldots) \boldsymbol{W}_m\right) \sigma'_k(\ldots) \boldsymbol{h}_{k-1}$

Structured Prediction

$p(y|x) = \frac{\exp(\mathrm{score}(y,x))}{Z(x)}$, $Z(x) = \sum_{y' \in \mathcal{Y}} \exp(\mathrm{score}(y', x))$
Problem: $\mathcal{Y}$ exponentially/infinitely large. **Solution**: Design algorithms using structure of input/output.

Language Modelling

$p(\boldsymbol{y}) = p(\mathrm{eos}|\boldsymbol{y}) \cdot \prod_{i=1}^{N} p(y_i \mid \boldsymbol{y}_{<i})$ $p(y_i|\boldsymbol{y}_{<i}) = \frac{1}{Z(\boldsymbol{y}_{<i})} \exp(\mathrm{score}(\boldsymbol{y}_{<i}, y_i))$
Non-tight: Force $p(\mathrm{eos}|\boldsymbol{y}_{<i}) > \xi > 0$
n-gram: $p(y_i|\boldsymbol{y}_{<i}) = p(y_i|y_{i-n+1}, \ldots, y_{i-1})$ **Neural n-gram**: Embeddings + MLP
RNN: $\boldsymbol{h}_i = \sigma(\boldsymbol{W}_h \boldsymbol{h}_{i-1} + \boldsymbol{W}_x \boldsymbol{e}(y_{i-1}) + \boldsymbol{b})$
Vanishing gradient: LSTM/GRU

Semirings

Definitions: **Monoid** $\langle \mathbb{K}, \circ, \boldsymbol{e} \rangle$: assoc, identity **Semiring** $\langle \mathbb{K}, \oplus, \otimes, \boldsymbol{0}, \boldsymbol{1} \rangle$: comm monoid, monoid, distrib, annihilator
Closed: $x^* = \bigoplus_{n=0}^{\infty} x^{\otimes n}$
Boolean, Viterbi $\langle [0,1], \max, \times, 0, 1 \rangle$, Inside, Real, Tropical, Log, Expectation, Counting

Part-of-Speech Tagging

Input: $\boldsymbol{w} \in \Sigma^N$. Output: $\boldsymbol{t} \in \mathcal{T}^N$.
CRF: $\mathrm{score}(\boldsymbol{t}, \boldsymbol{w}) = \sum_{n=1}^{N} \mathrm{score}(\langle t_{n-1}, t_n \rangle, \boldsymbol{w}, n) = \mathrm{trans}(t_{n-1}, t_n) + \mathrm{emit}(w_n, t_n)$
Forward: $\alpha_{n,t_n} \leftarrow \bigoplus_{t_{n-1} \in \mathcal{T}} \exp(\mathrm{score}(\ldots)) \otimes \alpha_{n-1,t_{n-1}}$ Return $\alpha_{N,\mathrm{eot}}$. Runtime: $O(N|\mathcal{T}|^2)$
Dijkstra: $O(N|\mathcal{T}|^2 + N|\mathcal{T}| \log(N|\mathcal{T}|))$

Finite-State Automata

WFST: $\Sigma, \Omega, Q, I \subseteq Q, F \subseteq Q, \lambda : I \to \mathbb{K}, \rho : F \to \mathbb{K}, \delta$
Pathsum: $Z(\mathcal{T}) = \bigoplus_{i,k \in Q} \lambda(q_i) \otimes \boldsymbol{R}_{ik} \otimes \rho(q_k)$
Lehmann: $\boldsymbol{R}_{ik}^{(j)} \leftarrow \boldsymbol{R}_{ik}^{(j-1)} \oplus \boldsymbol{R}_{ij}^{(j-1)} \otimes \left(\boldsymbol{R}_{jj}^{(j-1)}\right)^* \otimes \boldsymbol{R}_{jk}^{(j-1)}$ Runtime: $O(|Q|^3)$
Composition: $\mathcal{T}(x, y) = \bigoplus_{z \in \Omega^*} \mathcal{T}_1(x, z) \otimes \mathcal{T}_2(z, y)$

Transliteration

Map $\Sigma^* \to \Omega^*$. Three transducers:
1. $\mathcal{T}_x$: maps $x \to x$
2. $\mathcal{T}_{\boldsymbol{\theta}}$: maps $\Sigma^* \to \Omega^*$
3. $\mathcal{T}_y$: maps $y \to y$

Compose for $Z(x)$ and $\mathrm{score}(y, x)$

Constituency Parsing

CFG: $\mathcal{N}$, S, Σ, $\mathcal{R}$ (rules $N \to \boldsymbol{\alpha}$) **PCFG**: locally normalized. **WCFG**: globally normalized.
CNF: $N_1 \to N_2 N_3$ or $N \to a$ (no cycles)
CKY: $\boldsymbol{C}_{i,k,X} \leftarrow \bigoplus_{X \to YZ} \exp(\mathrm{score}(X \to YZ)) \otimes \boldsymbol{C}_{i,j,Y} \otimes \boldsymbol{C}_{j,k,Z}$ Return $\boldsymbol{C}_{1,N+1,S}$. Runtime: $O(N^3 |\mathcal{R}|)$

Dependency Parsing

$(N-1)^{N-2}$ spanning trees with single-root.
$\mathrm{score}(\boldsymbol{t}, \boldsymbol{w}) = \boldsymbol{\rho}_r + \sum_{(i \to j) \in \boldsymbol{t}} \boldsymbol{A}_{ij}$
Koo MTT: Laplacian $\boldsymbol{L}_{ij} = \begin{cases} \boldsymbol{\rho}_j \text{ if } i=1 \\ -\boldsymbol{A}_{ij} \text{ if } i \neq j \\ \sum_{k \neq i} \boldsymbol{A}_{kj} \text{ otherwise} \end{cases}$ $Z(\boldsymbol{w}) = \det(\boldsymbol{L})$. Runtime: $O(N^3)$
Chu-Liu-Edmonds: Greedy graph $\longrightarrow$ contract cycles $\longrightarrow$ swap loss $\longrightarrow$ expand. $O(N^2)$

Semantic Parsing

Lambda calculus: x, y, z; $(\lambda x. f(x))$; (MN) **β-reduction**: $((\lambda x.M)N) \to M[x := N]$ **α-conversion**: $\lambda x.M[x] \to \lambda y.M[y]$
CCG rules: $X/Y\ Y \Rightarrow X$ (>), $Y\ X \setminus Y \Rightarrow X$ (<) $X/Y\ Y/Z \Rightarrow X/Z$ ($\boldsymbol{B}_>$) $X \Rightarrow T/(T \setminus X)$ ($\boldsymbol{T}_>$)
LIG: CFG with stacks. Push/pop rules.

Transformers

Self-attention: Learn $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V \in \mathbb{R}^{d \times d}$ $\mathrm{SelfAtt}(\boldsymbol{X}) = \mathrm{softmax}\left((\boldsymbol{W}_Q^\top \boldsymbol{X})^\top \frac{\boldsymbol{W}_K^\top \boldsymbol{X}}{\sqrt{d_q}}\right)(\boldsymbol{W}_V^\top \boldsymbol{X})^\top$ Runtime: $O(nd^2 + dn^2)$
Positional encoding: $\boldsymbol{P}_{pi} = \sin(p/10000^{i/d})$ or cos
Encoder: $\oplus\ \boldsymbol{P} \to$ MHSA $\to \oplus \to$ LN $\to$ MLP $\to \oplus \to$ LN **Decoder**: + linear + softmax
Beam search, **nucleus sampling**

Axes of Modelling

Bias-variance: High bias = underfit. High variance = overfit. **Regularization**: $\ell(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_2^2$
MLE: $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} -\log \prod_{(x,y) \in \mathcal{D}} p_{\boldsymbol{\theta}}(y|x)$
Precision = TP/PP, **Recall** = TP/(TP+FN), **F1** = $2 \cdot \frac{P \cdot R}{P+R}$
Locally norm: efficient, label bias **Globally norm**: needs normalizer

Tips

Gradient: Sum over paths, product within paths. **Reuse terms** in backprop for efficiency.
Complexities: vec-vec $O(d)$, mat-vec $O(nm)$, mat-mat $O(nm\ell)$
Activations:
- $\sigma(x) = \frac{1}{1+\exp(-x)}$, $\sigma'(x) = \sigma(x)(1 - \sigma(x))$
- $\mathrm{ReLU}(x) = \max\{0, x\}$, $\mathrm{ReLU}'(x) = \mathbb{1}\{x > 0\}$
- $\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$, $\tanh'(x) = 1 - \tanh^2(x)$