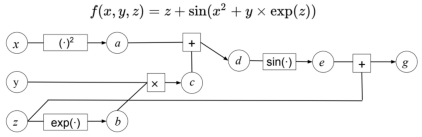
1 Backpropagation

Goal: Compute function gradient in the complexity of computing the function value.

Computational Graph with Hyperedges

$f(x,y,z) = z + \sin(x^2 + y \times \exp(z))$



2 Probability and Log-linear Models

A **Random Variable** is function from outcome space Ω to value space T (neither *random* nor *variable*). **Axioms of Probability**: (1) Non-negative, (2) Sums to 1, (3) $p(A \cup B) = p(A) + p(B) - p(A \cap B)$.

Log-linear Model

Def: $p(y|x,\theta) = \frac{\exp(\theta^\top \mathbf{f}(x,y))}{\sum_{y' \in \mathcal{Y}} \exp(\theta^\top \mathbf{f}(x,y'))}$.

MLE loss and its gradient:

$L(\theta) = \sum_n \log p(y_n|x_n,\theta)$

$\partial_{\theta_k} L(\theta) = \sum_n \left(f_k(x_n,y_n) - \mathbb{E}_{y' \sim \theta} f_k(x_n,y') \right)$.

Exponential Family

$p(x|\theta) = h(x)\exp(\theta^\top \phi(x))/Z(\theta)$ (canonical form), $\phi(x)$ minimum sufficient if it is independent. *Maximum Entropy Explanation*: exponential family maximizes $J(p) = -\sum_{x \in X} p(x)\log p(x)$ with constrains (1) non-negativity (2) sum to 1, (3) $\mathbf{F_k} = \sum_{x \in \mathcal{X}} p(x)\phi_k(x)$ for some func ϕ and const $\mathbf{F}$.

3 Word Embedding and Sentiment Analysis

Skip-gram Model (word2vec)

Preprocess: Given window size k and corpus size C, find all pairs with a focus word w and a context word c which locates in the window centered around the focus word. The complexity is $O(k \cdot C)$.

Model: $p(c \mid w) = \frac{\exp(e_{\text{wrd}}(w) \cdot e_{\text{ctx}}(c))}{Z(w)}$. Each word has two different embedding: e_{wrd} and e_{ctx}, either throw away e_{ctx} or concatenate it with e_{wrd} to form embeddings of dimensionality $2d$. Estimated via MLE.

Evaluation of embeddings

Word similarity: compare embedding similarity of similar words.

Word analogies: king - queen = man - woman.

Pipeline of Sentiment Analysis

Goal: Predict whether a sentence is positive or negative.

1. Preprocess: tokenization, stemming, stop word removal, etc.
2. Word embedding using some models.
3. Sentence embedding by pooling word embeddings: sum, mean, max, etc.
4. Run MLP on the sentence embedding which has a fixed size.

tf-idf= $\text{tf}(t,d) \times \text{idf}(t,D)$

tf: term frequency, how frequent is term t in document d,

idf: inverse document frequency, log(how frequent is term t in corpus D).

4 Language Modeling with n-gram & RNN

Goal: Model the distribution over all finite sequences from V^*, where V is the vocabulary set. The instance space is infinite.

Local Normalization

Def: the sum of the probability of all children given their parents is 1. This guarantees the normalizer is 1.

Necessary Condition: every node has "EOS" as descendant (could be child or grandchild, etc.), o.w. there is a possible sequence of infinite length.

Sufficient Condition: every node x has "EOS" as a child and $P(\text{EOS} \mid \text{x}) \geq c$ for some positive constant c.

n-gram Assumption

n-gram simplifies the problem by assuming each word only depends on the previous $n-1$ words, i.e., $p(y_t \mid y_{<t}) := p(y_t \mid y_{t-(n-1):t-1}) := \frac{\exp(w_{y_t} \cdot h_t)}{\sum_{y'} \exp(w_{y'} \cdot h_t)}$, where $w_y \in R^d$ is the word vector and h_t is the n-gram context vector which encodes the previous $n-1$ words. "BOS" is padded for some $t < n$. $h_t := f(y_{t-1},\ldots,y_{t-(n-1)})$ is computed by a neural network.

Recurrent Neural Network

Change $h_t := f(h_{t-1}, y_{t-1})$, where f is still a neural network. This allows information from arbitrary distance.

1. Vanilla RNN: $h_t = \sigma(W_1 h_{t-1} + W_2 e(y_{t-1}))$, $\sigma = \tanh$.
2. LSTM: has a short-term memory h_t and long-term memory c_t.
3. GRU: a fix to the vanishing/exploding gradients.

5 Part-of-Speech Tagging with CRF

Goal: Classify the grammatical categories of each word, e.g. noun, verb, etc. $p(t|w) = \exp(\text{score}(t,w))/Z(w)$ for a tagging sequence t and a sentence w of length N.

Conditional Random Field (a.k.a. Log-Linear Models on Structure)

Idea: additively decomposable score function: $\text{score}(w,t) = \sum_{n=1}^{N} \text{score}(< t_{n-1}, t_n >, w)$. The normalizer can be computed by DP (semiring: $(R^+ \cup \{+\infty\}, +, \times, 0, 1)$): $Z(w) = \beta(w, t_0)$, where

$\beta(\mathbf{w}, t_N) = \mathbf{1}$

for $n = N-1,\ldots,0$:

$\beta(\mathbf{w}, t_n) = \bigoplus_{t_{n+1} \in \mathcal{T}} \exp\{\text{score}\,\langle t_n, t_{n+1} \rangle, \mathbf{w})\} \otimes \beta(\mathbf{w}, t_{n+1})$

Decoding the best POS tagging

Find maximum-score path, using Viterbi (semiring: $([0,1], \max, \times, 0, 1)$).

Semiring

Def: (1) $(A, \oplus, \overline{0})$ commutative monoid, (2) $(A, \otimes, \overline{1})$ monoid, (3) $\otimes$ distributes over $\oplus$: $\forall a,b,c \in A$, $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$, $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$. (4) $\overline{0}$ annohilator for $\otimes$: $\forall a \in A$, $\overline{0} \otimes a = a \otimes \overline{0} = \overline{0}$.

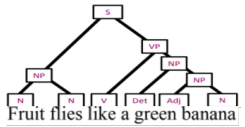
Semiring	Set	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$	intuition/application
Boolean	$\{0,1\}$	$\vee$	$\wedge$	0	1	logical deduction, recognition
Viterbi	$[0,1]$	max	$\times$	0	1	prob. of the best derivation
Inside	$\mathbb{R}^+ \cup \{+\infty\}$	$+$	$\times$	0	1	prob. of a string
Real	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$	0	shortest-distance
Tropical	$\mathbb{R}^+ \cup \{+\infty\}$	min	$+$	$+\infty$	0	with non-negative weights
Counting	$\mathbb{N}$	$+$	$\times$	0	1	number of paths

Structured Perceptron

The MLE loss for CRF is a softmax, if we add a temperature variable and take it to infinity, the loss becomes $\sum_i (\text{score}(t^i, w^i) - \max_{t'} \text{score}(t', w^i)) \Rightarrow$ structured perceptron.

6 Context-Free Parsing with CKY

Goal: Find the syntax parsing tree of a context-free grammar. A constituent is a word or a group of words that function as a single unit within a hierarchical structure. Every node in the parsing tree is a constituent.



Context-Free Grammar

Components: (1) Non-terminal symbol set, (2) a start non-terminal, (3) a set of terminal symbols and (4) a set of production rules. A grammar is in Chomsky normal form if all production rules are of the form: $N_1 \to N_2 N_3$, $N \to a$.

Probabilistic CFG

Each production rule has a probability. The probability of a tree is computed by multiplying the probabilities of all the edges. PCFG is locally normalized, making the normalizer 1.

Weighted CFG

Each production rule has a generic non-negative weight. The weights of trees are softmaxed to be a valid distribution. WCFG is globally normalized. In general, the output space is infinite, making computation of normalizer hard. However, we can efficiently compute the normalizer for a fixed sentence.

Parsing a string

$$p(t \mid s) = \exp(\text{score}(t))/Z(s)$$

To avoid divergence of $Z(s)$ which could be caused by the infinite number of parsing trees, we can only look at trees in CNF because the CNF theorem says that for any grammar G, there is another grammar G' that accepts the same set of strings and is in CNF. The number of possible CNF parsing tree (each non-terminal symbol has two children and each terminal symbol has one leaf child) is $O(4^N)$, no longer infinite.

CKY Algorithm

This algorithm finds all valid parsing trees under the given production rule set for CFG in CNF form in $O(N^3|R|)$, where N is the length of the string and R is the rule set.

WeightedCKY$(\mathbf{s}, \langle \mathcal{N}, S, \Sigma, \mathcal{R} \rangle, \text{score})$:

$N \leftarrow |\mathbf{s}|$

chart $\leftarrow 0$

for $n = 1,\ldots,N$:

for $X \to s_n \in \mathcal{R}$:

chart$[n, n+1, X]$ += $\exp\{\text{score}(X \to s_n)\}$

for $span = 2,\ldots,N$:

for $i = 1,\ldots,N - span + 1$:

$k \leftarrow i + span$

for $j = i+1,\ldots,k-1$:

for $X \to Y\,Z \in \mathcal{R}$:

chart$[i,k,X]$ += $\exp\{\text{score}(X \to Y\,Z)\} \times$ chart$[i,j,Y] \times$ chart$[j,k,Z]$

return chart$[1, N+1, S]$

Using the $(\max, \times)$ semiring decodes the best parsing.

7 Dependency Parsing (DP) using MTT

Goal

Construct dependency tree (DT) with syntactic relations, e.g., noun-modifier, determiner, etc. DT is a directed spanning(every node on graph connected) tree, with an additional root node having $\deg_{\text{out}} = 1$.

Relation to context free grammar

CFG: no information on syntactic relation; DP: no information on constituency structure.

Methods for cauculating Partition Function

(1) Projective DT: no crossing arcs. Equivalent to lexicalized CFG. Algorithms are generally dynamic programming. (2) Non-Projective DT: crossing arcs. Algorithms use matrix-tree theorem (MTT).

$\mathcal{Z}$ for Non-Proj DT in Edge-Factored Assump

Assume $p(\mathbf{t} \mid \mathbf{w}) = \frac{1}{Z} \prod_{(i \to j) \in \mathbf{t}} \exp\{\text{score}(i \to j, \mathbf{w})\} \exp\{\text{score}(\text{root}, \mathbf{w})\}$.

Def: edge score $A_{ij} = \exp(\text{score}(i,j,w))$, root score $\rho_j = \exp(\text{score}(j,w))$. **MTT** says $Z = |L|$ for following L which is $O(n^3)$:

$$L_{ij} = \begin{cases} -A_{ij} & \text{if } i \neq j \\ \rho_j + \sum_{k \neq i} A_{kj} & \text{otherwise} \end{cases}$$

Decoding the best DT

Find the best directed spanning tree starting from root and the degree of root is 1. Kruskal's algorithms (greedy) that work in undirected graph do not work.

Chu-Liu-Edmonds Algo ($O(n^3)$, optimized to $O(n^2)$ by Tarjan): (1) Find the best *incoming* edge for each vertex. (2) Contract cycles to be a single node c and increase the weight of incomming edge by $c.n_i$ with $w(c.n_i \to c.n_{i+1}) + \cdots$ until the last node in the circle. (3) Continue contraction until $\mathcal{G}$ has a spanning tree.

For each root edge in the MST of the contracted graph:

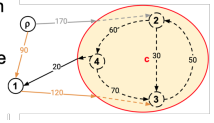
1. Find next best incoming edge to target node (c -> 1)
2. Calculate cost of removing edge (90 - 20 = 70)
3. Remember removal cost

Node	Cost
c	50
1	70

Consider deleting root edge to node with smallest removal cost

If deleting the edge would lead to a cycle in the greedy graph: Contract!

Otherwise: Delete edge



(4) Expand contract nodes by breaking cycles accordingly.

8 Semantic Parsing

Goal: Syntatic Representation => semantic representation.

Example: "Everybody loves someone else": (1) $\forall p.(\text{Person}(p) \to \exists q.(\text{Person}(q) \wedge p \neq q \wedge \text{Loves}(p,q)))$, (2) $\exists p.(\text{Person}(p) \wedge \forall q.(\text{Person}(q) \wedge p \neq q \to \text{Loves}(q,p)))$

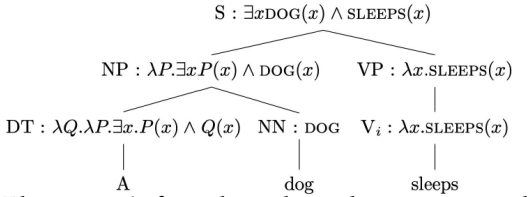
Principle of Compositionality

(Complex_expression A).meaning = f(A.constituents.meanings).

Enriched λ calculus to represent meanings

Computation rules: α-**conversion**: renaming, $\lambda x.\lambda y.(x((\lambda x.x\ x)y)) \to \lambda z.\lambda y.(z((\lambda x.x\ z)y))$.

β-**reduction**: applying one lambda term to another, $\lambda y.(z((\lambda x.x\ z)y)) \to \lambda y.(z(z\ y))$.



The semantic form depends on the grammar and combination order.

Derivation e.g.: "likes every dog".sem

= ("likes".sem "everydog".sem)

$= (\lambda P.\lambda Q.Q(\lambda x.P(\lambda y.\text{LIKES}(x,y)))) \quad \lambda Q.\forall x(\text{DOG}(x) \Rightarrow Q(x)))$

$= (\lambda P.\lambda Q.Q(\lambda t.P(\lambda y.\text{LIKES}(t,y)))) \quad \lambda Q.\forall x(\text{DOG}(x) \Rightarrow Q(x)))$

$= \lambda Q.Q(\lambda t.(\lambda Q.\forall x(\text{DOG}(x) \Rightarrow Q(x)) \quad (\lambda y.\text{LIKES}(t,y))))$

$= \lambda Q.Q(\lambda t.(\forall x(\text{DOG}(x) \Rightarrow ((\lambda y.\text{LIKES}(t,y))x))))$

$= \lambda Q.Q(\lambda t.(\forall x(\text{DOG}(x) \Rightarrow \text{LIKES}(t,x))))$

9 Transliteration with WFST

Def: Weighted finite-state transducers $T = \langle Q, \Sigma, \Omega, \lambda, \rho, \delta \rangle$ consist of: (1) Q a finite set of states (including initial or ending); (2) Σ, input vocab; (3) Ω, output vocab; (4) λ: $Q \to$initial scores; (5) ρ: $Q \to$final scores; (6) δ: arc $q_i \to q_j$ to transition scores. If no output vocab, WFST is a WFSA(acceptor).

Probabilistic model for transition sequences

π: path that generates the input X and output Y, $\text{score}(\pi) = \sum_i \text{score}(\pi_i)$. Assume $p(y \mid x) = \frac{1}{Z}\sum_\pi \exp(\sum_i \text{score}(\pi_i))$. **Goal**: find the min/max score path.

Floyd-Warshall Algo $O(N^3)$ (semi ring version)

```
let dist be a N × N array of minimum distances initialized to 0 (infinity)
for each edge (u, v) do
    dist[u][v] ← W[u][v]  // This corresponds to W¹
for each vertex v do
    dist[v][v] ← W[v][v] // This corresponds to W⁰
for k from 1 to N
    for i from 1 to N
        for j from 1 to N
            dist[i][j] ←dist[i][j] ⊕ (dist[i][k] ⊗ dist[k][k]* ⊗ dist[k][j])
```

Shortest Path: Real semiring. **Partition function** $\mathcal{Z}$: Inside semiring.

Kleene's Star r^*

Inside semiring	$I + W^1 + W^2 + W^3 + \ldots$	$r^* = 1/(1-r)$ (if $r \neq 1$; otherwise $r^* = \infty$)
Counting semiring	$I + W^1 + W^2 + W^3 + \ldots$	$r^* = \infty$
Tropical semiring	min(0, W, 2W, 3W, …)	$r^* = 0$
Viterbi semiring	max(I, W^1, W^2, W^3…)	$r^* = 1$
Boolean semiring	OR(True, A, A AND A,…)	0* = 1* = True

Normalizer computation

α: starting weight; β ending weights; W^ω weight matrix for arc ω. Partitoin function $Z = \alpha^T(\sum_{\omega \in \Omega \cup \{\epsilon\}} W^\omega)^* \beta$, where the $*$ (Kleene closure) is computed by FW algo with Inside semiring.

10 Machine Translation with Transformer

Difference from Transliteration

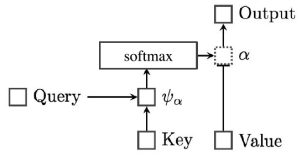
(1) many correct translations. (2) locality assumption (scores = $\sum_{\text{arcs}} \text{score}_i$) not reasonable.

Seq2Seq Model

A representation method: $z = \text{encoder}(x)$ and $y \mid x \sim \text{decoder}(z)$. Local normalization: $p(y \mid x) = \prod_t p(y_t \mid x, y_1, \ldots, y_{t-1})$. Iteratively predict on $p(y_t \mid x, y_1, \ldots, y_{t-1})$.

For standard RNN, the decoder only receives one vector, causing information bottleneck.

Attention Mechanism



$Q = W^Q X_Q, K = W^K X_K, V = W^V X_V,$

$\text{context}(Q,K,V) = \text{softmax}\left(QK^T/\sqrt{d_k}\right)V$

Standard attention: $X_K = X_V = h^{\text{en}}, X_Q = h^{\text{dec}}$.

Self-attention: $X_K = X_V = X_Q = h^{\text{en}}$ or h^{dec}.

Decoding

Beam Search: pruned breadth-first search where the breadth is limited to size k.

Sampling: sample according to the conditional distribution at each time step.

11 Axes of Modeling

Modeling Choosing

- **Probabilistic**(Logistic reg) *pros*: can utilize prob theory for learning; *cons*: assumption (independence or noise distribution) unrealistic. **Non-probabilistic**(learned:SVM, perceptron, manual:CFG) *pros*: interpretable *cons*: not straightforward to quantify uncertainty.
- **Generative** (N-gram, Markov RF, RNN): model the distribution of each class; **Discriminative**(CRF, RNN): model the decision boundary between classes.
- **Locally Normalized** vs **Globally**.

Model selection

Use nested CV to:(1) prevent data leakage (2) check model stability.

12 Bias and Fairness

Goal

Consider bias in the labeling, sample selection, task, features, loss functions and feedback loops.

Train-Test Mismatch

Problem: distri mismatch btw old & new data.

Unsupervised Adaption (no label of new data)

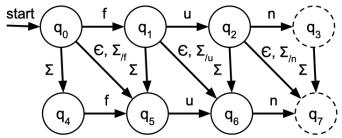
Importance sampling: $\mathbb{E}_{(x,y)\sim\mathcal{D}^{\text{new}}} l(y, f(x)) = \mathbb{E}_{(x,y)\sim\mathcal{D}^{\text{old}}}\left(\frac{\mathcal{D}^{\text{new}}(x,y)}{\mathcal{D}^{\text{old}}(x,y)} l(y, f(x))\right)$. Since $\frac{\mathcal{D}^{\text{new}}(x,y)}{\mathcal{D}^{\text{old}}(x,y)}$ is unknown, we could use approximations: $\mathcal{D}^{\text{new}} \propto \mathcal{D}^{\text{base}} p(s=0 \mid x)$ and $\mathcal{D}^{\text{old}} \propto \mathcal{D}^{\text{base}} p(s=1 \mid x)$. Therefore, $\frac{\mathcal{D}^{\text{new}}(x,y)}{\mathcal{D}^{\text{old}}(x,y)} = \frac{1}{p(s=1|x)} - 1$. We can train a classifier to distinguish between old and new distribution and thus get the estimation of $p(s=1 \mid x)$.

Supervised Adaption (some labels of new data)

Use feature augmentation (separation) by forcing features into three components: shared, old-only (forced to be zero in the new dataset) and new-only (forced to be zero in the old dataset).

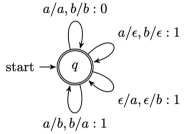
A Examples of WFSA/T

1. FSA for at most 1 edit for word *fun*, to expand to d edit for words of length N, expand this $1 \times N$ graph to $d \times N$.

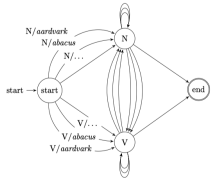


2. rules and WFST for arbitrary words edit distance:

$$\delta(q,a,a,q) = \delta(q,b,b,q) = 0$$
$$\delta(q,a,b,q) = \delta(q,b,a,q) = 1$$
$$\delta(q,a,\epsilon,q) = \delta(q,b,\epsilon,q) = 1$$
$$\delta(q,\epsilon,a,q) = \delta(q,\epsilon,b,q) = 1.$$



3. WFST for HMM with labels $\{N, V\}$:



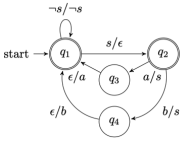
4. WFST for transforming plural form to original:

-sses → -ss e.g., *dresses* → *dress*

-ies → -i e.g., *parties* → *parti*

-ss → -ss e.g., *dress* → *dress*

-s → ϵ e.g., *cats* → *cat*



B GRU and LSTM