I. Backpropagation

1. Log (ln) and Exp Operations

$e^0 = 1$, $e^{-\infty} \to 0$, $\ln 1 = 0$, $\ln e = 1$.

$e^x \cdot e^y = e^{x+y}$	$\ln(x \cdot y) = \ln x + \ln y$
$e^x / e^y = e^{x-y}$	$\ln(x/y) = \ln x - \ln y$
$(e^x)^y = e^{xy}$	$\ln(x^y) = y \ln x$
$e^{\ln x} = x$	$\ln(e^x) = x$

II. Log-Linear Models

1. Exponential Family

$p(x|\theta) = \frac{1}{Z(\theta)} h(x) e^{\theta \cdot \phi(x)}$, where $Z(\theta)$ is partition function, $h(x)$ determines supports, θ is canonical parameters, $\phi(x)$ is sufficient statistics, finite.

2. Log-Linear Models

$p(y|x,\theta) = \frac{1}{Z(\theta)} e^{\theta \cdot f(x,y)}$, $x \in X$, $y \in Y$, feature $f: X \times Y \to \mathbb{R}^K$, parameters $\theta \in \mathbb{R}^K$. $Z(\theta) = \sum_{y' \in Y} e^{\theta \cdot f(x,y')}$, $O(|Y|)$ computation.

3. Softmax

$\text{softmax}(h, y, T) = \frac{e^{h_y/T}}{\sum_{y' \in Y} e^{h_{y'}/T}}$, $h_y = \theta \cdot f(x,y)$, temperature $T \in \mathbb{R}$, $T \to \infty$ uniform, $T \to 0$ argmax (annealing).

III. Multilayer Perceptron (MLP)

1. Multilayer Perceptron (MLP)

$h^{(N)} = \sigma^{(N)}(W^{(N)} \dots \sigma^{(2)}(W^{(2)} \sigma^{(1)}(W^{(1)} e(x))))$, $h^{(N)} \in \mathbb{R}^{|Y|}$, activation $\sigma^{(i)}$, $W^{(N)} \in \mathbb{R}^{|Y| \cdot d_N}$, $W^{(1)} \in \mathbb{R}^{d_1 \cdot d_1}$, encoding $e(x) \in \mathbb{R}^{d_1}$. Then, MLP is $p(y|x) = \frac{\exp(h_y)}{\sum_{y' \in Y} \exp(h_{y'})} = \text{softmax}(h^{(N)}, y)$.

MLP is a log-linear model, where we also learn the feature f. Final layer is a softmax.

2. XOR Problem $y = \alpha_1 x_1 + \alpha_2 x_2 + b$

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Not linearly separable (a single-layer MLP can't solve). Use activations: $\tanh(x) = 2\sigma(2x) - 1$ or sigmoid $\sigma(x) = \frac{1}{1+\exp(x)}$.

IV. Language Models: n-grams and RNNs

1. Language Modeling

Alphabet Σ is a finite, non-empty set of symbols. A **string** over Σ is finite sequence of alphabet symbols. **Kleene closure** Σ^* is the set of all possible strings.

2. Globally Normalized Language Models

$p(\boldsymbol{y}) = \frac{1}{Z} e^{\text{score}(\boldsymbol{y})}$, $Z = \sum_{y' \in \Sigma^*} e^{\text{score}(y')}$ is the normalization constant, infinite sum, **not always computable**; score: $\boldsymbol{y} \to \mathbb{R}$.

3. Locally Normalized Language Models

With $\boldsymbol{y} = y_1 y_2 \dots y_N$ and $\boldsymbol{y}_{<N} = y_1 y_2 \dots y_{N-1}$, $p(\boldsymbol{y}) = p(y_1|\text{BOS}) p(y_2|\text{BOS } y_1) \dots p(y_N|\boldsymbol{y}_{<N}) p(EOS|\boldsymbol{y})$.

- Local normalization **guarantees** the **normalization constant** to be **1**.
- The **sum of the probability** of all children **given their parent is 1**.
- Every node has an **EOS** as a descendant.

4. Tightness

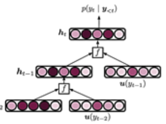
- **A locally normalized LM that sums to 1 is called tight**.
- A non-tight loses probability to infinitely long structures - **sequence models**.
- To ensure tightness, force $p(\text{EOS}|\text{parent}) > \xi > 0$ for every parent node with constant ξ.

5. n-gram Language Models

Assumption: limit the context to the previous $\boldsymbol{n-1}$ symbols. A **finite** number of histories. $p(y_t|\boldsymbol{y}_{<t}) = p(y_t|y_{t-n+1} \dots y_{t-1})$, $\boldsymbol{y} \in \Sigma^*$, $p(\boldsymbol{y}) = p(\text{EOS}|y_{t-n+2} \dots y_t) \prod_{t=1}^{T} p(y_t|y_{t-n+1} \dots y_{t-1})$.

6. Recurrent Neural Network (RNN)

$p(y_t|\boldsymbol{y}_{<t}) = \frac{e^{u(y_t) \cdot h_t}}{\sum_{y' \in \Sigma} e^{u(y'_t) \cdot h_t}}$, $u(y_t)$ is word embedding - individual symbols, h_t is context embedding - summarizes $n-1$ symbols, f is RNN type.



7. Vanilla / Elman RNN

Elman: $\boldsymbol{h}_t = \sigma(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{V}u(y_{t-1}) + \boldsymbol{b}_h)$

Variant: $\boldsymbol{h}_t = \sigma(\boldsymbol{W}[\boldsymbol{h}_{t-1}; u(y_{t-1})])$

$\boldsymbol{W} \in \mathbb{R}^{d \times 2d}$, $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{d \times d}$ are recurrence matrices, σ is a non-linearity as in an MLP.

- Trained with backpropagation through **time (temporal hidden-state dependencies)**.
- Each timestamp yields an **output** and a **recurrent connection**.
- Parameters are **shared** across timestamps.
- Unroll RNN first, then backpropagate.

8. LSTM, GRU, Vanishing / Exploding

Vanishing gradient - update < 1, **exploding gradient** - update > 1. **LSTM** and **GRU** can help solve the **vanishing gradient problem** as they have **cell state / gate update** with **additive** update. **ReLU** also works. **Sigmoid** and **Tanh** can lead to vanishing gradient.

V. Part-of-speech Tagging with CRFs

1. Conditional Random Fields (CRF)

$p(\boldsymbol{t}|\boldsymbol{w}) = \frac{\exp\{\text{score}(\boldsymbol{t},\boldsymbol{w})\}}{\sum_{t' \in T^N} \exp\{\text{score}(\boldsymbol{t}',\boldsymbol{w})\}}$, $\text{score}(\boldsymbol{t},\boldsymbol{w}) = \sum_{n=1}^{N} \text{score}(\langle t_{n-1}, t_n \rangle, \boldsymbol{w})$, $\boldsymbol{t}$ is part of speech tagging, $\boldsymbol{w}$ is an input sentence, $N = |\boldsymbol{w}|$.

$\sum_{t_1 \in T} \exp\{\text{score}(\langle t_0, t_1 \rangle, \boldsymbol{w})\} \times \left(\sum_{t_2 \in T} \exp\{\text{score}(\langle t_1, t_2 \rangle, \boldsymbol{w})\} \times \dots \times \left(\sum_{t_N \in T} \exp\{\text{score}(\langle t_{N-1}, t_N \rangle, \boldsymbol{w})\}\right)\right)$. Score can be chosen, consisting of **transition** (how likely t_2 follows t_1) and **emission** (how likely current word is t_2). Combinatorial assumption

2. Viterbi Algorithm (for shortest path)

```
def ViterbiAlgorithm(w,T,N):
  for t_{N-1} ∈ T:
    v(w, t_{N-1}, N-1) ← e^{score(⟨t_{N-1},EOS⟩,w)}
  for n ∈ N-2, ...,1:
    for t_n ∈ T:
      v(w, t_n, n) ← max_{t_{n+1}∈T} e^{score(⟨t_n,t_{n+1}⟩,w)} × v(w, t_{n+1}, n+1)
      b(t_n, n) ← argmax_{t_{n+1}∈T} e^{score(⟨t_n,t_{n+1}⟩,w)} × v(w, t_{n+1}, n+1)
 v(w, BOS, 0) ← max_{t_1∈T}(v(w, BOS, 0), e^{score(⟨BOS,t_1⟩,w)} × v(w, t_1, 1))
 b(BOS, 0) ← argmax_{t_1∈T}(v(w, BOS, 0), e^{score(⟨BOS,t_1⟩,w)} × v(w, t_1, 1))
 for n ∈ 1, ..., N:
   t_n ← b(t_{n-1}, n-1)
 return t_{1:N}, v(w, BOS, 0)
```

Replacing max with sum is **Backward Algo**.

The b in Viterbi is the backpointer for the best scoring path. Overall complexity $O(N|T|^2)$. Can **generalize** the algorithm with **semirings**.

3. Semirings

A semiring $R = (A, \oplus, \otimes, \bar{0}, \bar{1})$ must satisfy:

- $(A, \oplus, \bar{0})$ is a **commutative monoid**;
- $(A, \otimes, \bar{1})$ is a **monoid**;
- $\otimes$ **distributes** over $\oplus$: $\forall a, b, c \in A$, $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$, $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$;
- $\bar{0}$ is **annihilator** of $\otimes$: $\bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$.

VI. Context-Free Parsing with CKY

1. Context-Free Grammar (CFG)

A context-free grammar G is a quadruple $\langle \mathcal{N}, S, \varepsilon, \mathcal{R} \rangle$ consisting of:

- A finite set of **non-terminal** symbols $\mathcal{N}$;
- A distinguished start non-terminal symbol S;
- An alphabet of **terminal** symbols Σ;
- A set of production rules $\mathcal{R}$ of the form $N \to \alpha$, where $N \in \mathcal{N}$ and $\alpha \in (\mathcal{N} \cup \Sigma)^*$.

2. Probabilistic CFGs (PCFG)

$p(\text{tree}) = \prod_{N \in \mathcal{N}, \alpha \in (\mathcal{N} \cup \Sigma)^*} p(N \to \alpha)$. PCFGs are **locally normalized**. For all rules with the same left-hand side, e.g., $N \to \alpha_1, \dots, N \to \alpha_k$, the **sum of probability must be 1**.

3. Weighted CFGs (WCFG)

$\exp\{\text{score}(\text{tree})\} = \prod_{N \in \mathcal{N}, \alpha \in (\mathcal{N} \cup \Sigma)^*} \exp\{\text{score}(N \to \alpha)\}$. WCFGs are globally normalized, i.e., $p(\boldsymbol{t}) = \frac{1}{Z} \prod_{r \in t} \exp\{score(r)\}$, $Z = \sum_{t' \in T} \prod_{r' \in t} \exp\{score(r')\}$, T is countably infinite.

4. Chomsky Normal Form (CNF)

A grammar is in CNF is all productions have the form: **(1)** $N_1 \to N_2 N_3$, $N_{1,2,3}$ are non-terminals; **(2)** $N \to \alpha$, N is a non-terminal, and α is a terminal; **(3)** $S \to \varepsilon$, S is start symbol and ε is empty string. With CNF, we can partition the **WCFG** into **non-terminal production** and **terminal production**.

5. Cocke-Kasami-Younger (CKY)

CKY requires CNF

```
def SemiringCKY(s,⟨N,S,Σ,R⟩,score):
  N ← |s|
  chart ← 0
  for n = 1, ..., N:
    for X → s_n ∈ R:
      chart[n, n+1, X] ⊕= e^{score(X→s_n)}
  for span = 2, ..., N:
    for i = 1, ..., N - span + 1:
      k ← i + span
      for j = i + 1, ..., k - 1:
        for X → Y Z ∈ R:
          chart[n, n+1, X] ⊕= e^{score(X→Y Z)} ⊗ chart[i, j, Y] ⊗
chart[j, k, Z]
  return chart[n, N+1, S]
```

Replacing $\oplus$ with $+$ and $\otimes$ with $\times$ will give us the weighted CKY. Complexity $O(N^3|\mathcal{R}|)$, N is sentence length, $|\mathcal{R}|$ is rule set size.

VII. Dependency Parsing with MTT

1. Dependency Trees

(1) Projective: no crossing arcs, related to constituency. **(2) Non-projective**: crossing arcs, related to discontinuous constituency.

2. Distributions Over Non-projective Trees

$p(\boldsymbol{t}|\boldsymbol{w}) = \frac{1}{Z} e^{\text{score}(\boldsymbol{t},\boldsymbol{w})}$, $Z = \sum_{\boldsymbol{t}' \in T(\boldsymbol{w})} e^{\text{score}(\boldsymbol{t}',\boldsymbol{w})}$, score presents the compatibility of the parse $\boldsymbol{t}$ with sentence $\boldsymbol{w}$, $T(\boldsymbol{w})$ is all admissible parses of sentence $\boldsymbol{w}$, $N = |\boldsymbol{w}|$ input sentence length. Computing Z requires $O(N^N)$, spanning trees N^{N-2}, root constraint $(N-1)^{N-2}$. N^{N-1} for **directed graphs**, e.g., dependency parsing.

3. Edge-factored Assumption

$\text{score}(\boldsymbol{t},\boldsymbol{w}) = \sum_{(i \to j) \in \boldsymbol{t}} \text{score}(i \to j, \boldsymbol{w}) + \text{score}(r, \boldsymbol{w})$, where r is the root according to the tree $\boldsymbol{t}$. Edges are the first part of the sum. Probability $p(\boldsymbol{t}|\boldsymbol{w}) = \frac{1}{Z} \prod_{(i \to j) \in \boldsymbol{t}} e^{\text{score}(i,j,\boldsymbol{w})} e^{\text{score}(r,\boldsymbol{w})}$, $Z = \sum_{\boldsymbol{t}' \in T(\boldsymbol{w})} \prod_{(i \to j) \in \boldsymbol{t}'} e^{\text{score}(i,j,\boldsymbol{w})} e^{\text{score}(r,\boldsymbol{w})}$.

4. Matrix-Tree Theorem (MTT) $O(N^3)$

Let $A_{ij} = e^{\text{score}(i,j,\boldsymbol{w})}$, $\rho_j = e^{\text{score}(j,\boldsymbol{w})}$, $N_T(G) = |\hat{L}_i|$. **(1) Graph Laplacian**: $L_{ij} = -A_{ij}$ if $i \neq j$, $\sum_{k \neq i} A_{kj}$ otherwise. **(2) Modified Graph Laplacian**: ρ_j if $i = 1$ (root), $-A_{ij}$ if $i \neq j$, $\sum_{k \neq i} A_{kj}$ otherwise. Now $Z = |L| = \det(L)$.

5. Chu-Liu-Edmonds Algorithm $O(N^3)$

To find the best parse of a sentence (maximum-weight spanning tree - MST), $\text{argmax}_{\boldsymbol{t} \in T} \sum_{(i \to j) \in \boldsymbol{t}} \text{score}(i, j, \boldsymbol{w})$.

```
def MST(G):
  if CYCLE IN GREEDY(G):
    return
EXPAND(CONSTRAIN(MST(CONTRACT(G,CYCLE))))

def CONSTRAIN(G):
  if NUMBER OF ROOT EDGES(GREEDY(G)) > 1:
    e ← ROOT EDGE TO REMOVE (G)
    if CYCLE IN GREEDY(G - e):
      return CONSTRAIN(CONTRACT(G,CYCLE))
    else:
      return CONSTRAIN(G - e)
  else:
    return GREEDY(G)
```

For a cycle C, we have **(1) exit edges** emanating from C, **(2) enter edges** pointing to C, **(3) dead edges** inside or both ends in C, **(4) external edges** are outside C.

VIII. Semantic Parsing with CCG

1. Principle of Compositionality

The meaning of a complex express is a function of the **meanings** of that expression's constituent parts.

2. Lambda Calculus

If M is a term, x is a variable, $\lambda x.M$ is a term, which takes x as input and produces M. Scope: $((\lambda x.\lambda y.(x((\lambda x.x\,x)y))\lambda x.x)z)$.

(1) $\boldsymbol{\alpha}$-conversion

Renaming a variable in a lambda term, together with all occurrences, e.g., $\lambda x.\lambda y(x((\lambda x.x\,x)y)) \to \lambda z.\lambda y(z((\lambda x.x\,z)y))$.

(2) $\boldsymbol{\beta}$-reduction

Applying one lambda term to another, e.g., $\lambda y.(z((\lambda x.x\,z)y)) \to \lambda y.(z(z\,y))$.

Warning: repeatedly applying β-reductions may not terminate $(F\,F) \to (\lambda x.((x\,x)F)F) \to ((F\,F)F) \to (((F\,F)F)F) \to \cdots$.

(3) Logical constants

- **Objects & relations**: ALEX, MOSKVA, LIKES, TEACHER, etc.
- **Arity of relations**: LIKES(x, y) has arity 2, TEACHER(x) has arity 1, etc.

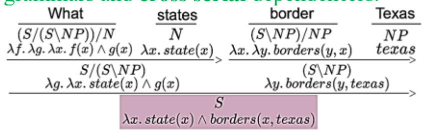
(4) Variables: uppercase $(P, Q$, etc.) for relations, lowercase $(x, y$, etc.) for objects.

(5) Literals: Applying relations to objects or variables, e.g., LIKES(LEE, BOB), TEACHER(BOB), LIKES(LEE, y), P(LEE, BOB), P(x, y), etc.

Note: With these we can construct logical terms with logical connectives and quantifiers. Can also form lambda terms.

3. Combinatory Categorical Grammars

Use CCG to deal with **context-sensitive** grammars and **cross serial** dependencies.



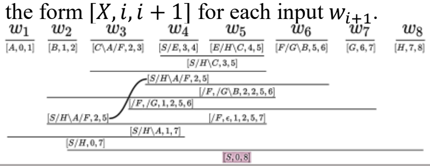
(1) Definition: A CCG is $\langle V_T, V_N, S, f, R \rangle$, where V_T is finite set of **terminals** (lexicon), V_N is finite site of **non-terminals** (atomic categories), $S \in V_N$ a distinguished category, f maps $V_T \cup \{\varepsilon\}$ to **finite** subsets of $C(V_N)$, set of categories, R is **combinatory rules**.

(2) Combinatory rules: forward (x/y) is $y \to x$, backward y is $(x \backslash y) \to x$.

Note: **CCG** in **higher-order** composition rules, each rule may give **infinite instances**. **CFGs** have a **finite** set of non-terminals.

4. Parsing CCGs (CKY style)

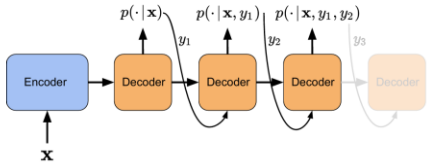
One inference rule for every forward rule $\frac{[X/Y, i, j][Y\beta, j, k]}{X\beta, i, k}$, $X/Y\,Y\beta \Rightarrow X\beta$. Axioms have the form $[X, i, i+1]$ for each input w_{i+1}.



IX. Machine Translation Transformers

1. Sequence-to-sequence Models

Model the probability distribution $p(\boldsymbol{y}|\boldsymbol{x})$ over all strings $\boldsymbol{y} \in Y$ for some sentence $\boldsymbol{x}$, i.e., what is the most likely translation $\boldsymbol{y}$ of string $\boldsymbol{x}$. Maximizing the log-likelihood $\text{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{N} \log p(\boldsymbol{y}^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{\theta}) = \text{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sum_{t=1}^{|\boldsymbol{y}^{(i)}|} \log p(y_t^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{y}_{<t}^{(i)}; \boldsymbol{\theta})$.



2. The Attention Mechanism

(1) Definition

The attention mechanism enables a model to attend to information from different time steps, $\boldsymbol{\alpha} = \mathrm{softmax}(\mathrm{score}(\boldsymbol{q}, K))$, $\boldsymbol{c} = \boldsymbol{\alpha} V$, where $\boldsymbol{q}$ is the query, K is the keys, V is the values, $\boldsymbol{c}$ is the resulting context.

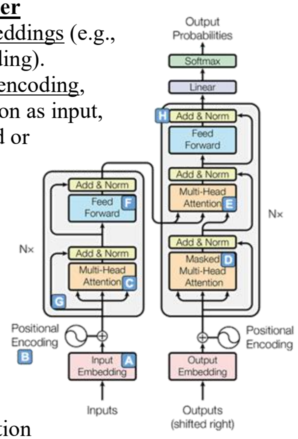
(2) Variations of Attention Mechanisms

(i) Cross-attention *Without projection*: $\boldsymbol{q_t} = \boldsymbol{h_t^d}$; $\boldsymbol{k_i} = \boldsymbol{v_i} = \boldsymbol{h_i^e}$, $i \in 1 \cdots n$; $K = V = H^e$.
With linear projection $(W_q, W_k, W_v \in \mathbb{R}^{d \times d})$: $\boldsymbol{q_t} = \boldsymbol{h_t^d} \times W_q^d$; $\boldsymbol{k_i} = \boldsymbol{h_i^e} \times W_k^e$; $\boldsymbol{v_i} = \boldsymbol{h_i^e} \times W_v^e$; $K = H^e \times W_k^e$; $V = H^e \times W_v^e$; $i \in 1 \cdots n$

(ii) Self-attention *Without projection*: $\boldsymbol{q_t} = \boldsymbol{h_t^s}$; $\boldsymbol{k_i} = \boldsymbol{v_i} = \boldsymbol{h_i^s}$, $i \in 1 \cdots n/m$; $K = V = H^s$. *With linear projection*: $\boldsymbol{q_t} = \boldsymbol{h_t^s} \times W_q^s$; $\boldsymbol{k_i} = \boldsymbol{h_i^s} \times W_k^s$; $\boldsymbol{v_i} = \boldsymbol{h_i^s} \times W_v^s$; $K = H^s \times W_k^s$; $V = H^s \times W_v^s$; $i \in 1 \cdots n$; $s \in \{e, d\}$.

Note: n is input length; m is output length (in self-attention **decoder**); $\boldsymbol{h}$ is the hidden state, e is encoder, d is decoder; $\boldsymbol{q_t}, \boldsymbol{k_i}, \boldsymbol{v_i}, \boldsymbol{h_i^e}, \boldsymbol{h_t^d} \in \mathbb{R}^{1 \times d}$; $H^e \in \mathbb{R}^{n \times d}$; $H^d \in \mathbb{R}^{m \times d}$.

3. Transformer



A: word embeddings (e.g., one-hot encoding).
B: positional encoding, same dimension as input, can be learned or fixed. $\sin p/C^{i/d}$ if $i = 2k$, $\cos p/C^{i/d}$ if $i = 2k+1$
C: self-attention
D: masked self-attention, in *decoder*, position after current time stamp cannot be attended.
E: cross-attention, allows decoder to attend encoder.
F: feed-forward layers, linear projections followed by non-linearities.
G: residual connection, in both encoder and decoder, passes input to next layer without transformation, help with vanishing gradients.
H: layer normalization, mean 0, variance 1.

4. Decoding Strategies

$O(|\Sigma|^n)$ due to non-markovian structure $\boldsymbol{y}_{<t}$.
(1) Beam search (TopK): Pruned breadth-first search where the breadth is limited to size k. Maximum of k paths kept at each time step. Greedy, no guarantee.
(2) Sampling (TopP): Sample according to the conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$ at each time step. Sample only from top items that cover p% of probability mass.

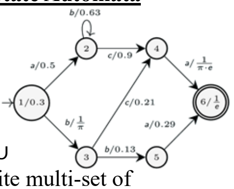
X. Transliteration with WFSTs

1. Finite-State Automata

Determines if a string is an element of a given language. A FSA $\mathcal{A}$ is a 5-tuple $(\Sigma, Q, I, F, \delta)$ where Σ is **alphabet**, Q is a finite set of **states**, $I \subseteq Q$ is the set of **initial states**, $F \subseteq Q$ is the set of **final** or **accepting states**, $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ is a finite multi-set.
Unambiguous if for every string $s \in \Sigma^*$ there is **at most 1 accepting path** for that s.
Note: Vertices are the states in Q, edges are transitions in δ, edge labels correspond to input symbol in Σ.

2. Weighted Finite-State Automata

A WFSA $\mathcal{A}$ over a semiring $\mathcal{W} = (\mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ is $(\Sigma, Q, I, F, \delta, \lambda, \rho)$ where in addition to FSA, $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times \mathbb{K} \times Q$ is a finite multi-set of **transitions**, $\lambda: Q \to \mathbb{K}$ an **initial weighting function** over Q, $\rho: Q \to \mathbb{K}$ a **final weighting function** over Q, $I = \{q \in Q | \lambda(q) \neq \mathbf{0}\}$ and $F = \{q \in Q | \rho(q) \neq \mathbf{0}\}$.

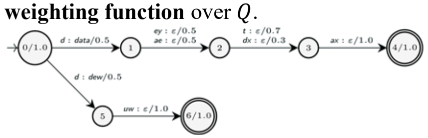


3. Path

A path $\boldsymbol{\pi}$ is an element of δ^* with consecutive transitions $(q_1 \to^{\cdot/\cdot} q_2, \cdots, q_{n-1} \to^{\cdot/\cdot} q_N)$. $p(\boldsymbol{\pi}) = q_1$ is the origin, $n(\boldsymbol{\pi}) = q_N$ is the destination. The **length** is the number of transitions $|\boldsymbol{\pi}|$. The **yield** of a path is the concatenation of the input symbols on the edges along the path $s(\boldsymbol{\pi})$. A path $\boldsymbol{\pi}$ is a cycle if the starting and ending states are the same.

4. Weighted Finite-State Transducers

A WFST $\mathcal{T}$ over a semiring $\mathcal{W} = (\mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ is $(\Sigma, \Omega, Q, I, F, \delta, \lambda, \rho)$ where Σ is finite **input alphabet**, Ω is finite **output alphabet**, Q is finite set of **states**, $I \subseteq Q$ is **initial states**, $F \subseteq Q$ is **final states**, $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Omega \cup \{\varepsilon\}) \times \mathbb{K} \times Q$ finite **multi-set of transitions**, $\lambda: Q \to \mathbb{K}$ **initial weighting function** over Q, $\rho: Q \to \mathbb{K}$ **final weighting function** over Q.



5. Composition of WFSTs

Composition $\mathcal{T}_1 \circ \mathcal{T}_2$ of two WFSTs $\mathcal{T}_1 = (\Sigma, \Omega, Q_1, I_1, F_1, \delta_1, \lambda_1, \rho_1)$ and $\mathcal{T}_2 = (\Sigma, \Theta, Q_2, I_2, F_2, \delta_2, \lambda_2, \rho_2)$ is the WFST $\mathcal{T} = (\Sigma, \Theta, Q, I, F, \delta, \lambda, \rho)$ such that $\mathcal{T}(\boldsymbol{x}, \boldsymbol{y}) = \bigoplus_{\boldsymbol{z} \in \Omega^*} \mathcal{T}_1(\boldsymbol{x}, \boldsymbol{y}) \otimes \mathcal{T}_2(\boldsymbol{z}, \boldsymbol{y})$.

6. Pathsum

$\mathcal{A}$ be a WFSA over a semiring $\mathcal{W} = (\mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$. The pathsum in $\mathcal{A}$ is defined as $Z(\mathcal{A}) = \bigoplus_{\boldsymbol{\pi} \in \Pi(\mathcal{A})} w(\boldsymbol{\pi})$. **Pathsum** between two states $q_n, q_m \in Q$ as $Z(q_n, q_m) = \bigoplus_{\boldsymbol{\pi} \in \Pi(q_n, q_m)} w(\boldsymbol{\pi})$. **Inner path weight** $w_I(\boldsymbol{\pi})$ of a path $\boldsymbol{\pi} = q_0 \to^{a_1/w_1} q_1 \cdots q_{N-1} \to^{a_N/w_N} q_N$ is $w_I(\boldsymbol{\pi}) = \bigoplus_{n=1}^{N} w_n$. $w(\boldsymbol{\pi})$ is the **path weight** as $w(\boldsymbol{\pi}) = \lambda(p(\boldsymbol{\pi})) \otimes w_I(\boldsymbol{\pi}) \times \rho(n(\boldsymbol{\pi}))$.

7. WFST Log-Linear Model

$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z} e^{\{\mathrm{score}(\boldsymbol{y}, \boldsymbol{z})\}} = \frac{1}{Z} \sum_{\boldsymbol{\pi} \in \Pi(x,y)} e^{\sum_{n=1}^{|\boldsymbol{\pi}|} \mathrm{score}(\tau_n)}$, where $Z = \sum_{y' \in \Omega^*} e^{\{\mathrm{score}(y', x)\}}$, needs algorithms.

8. Lehmann's Algorithm $O(N^3)$

```
def Lehmann(W):
  W be a N×N array of minimum distance 0
  for each edge (u,v):
    W[u][v] ← W[u][v]
  for each vertex v:
    W[v][v] ← W[v][v]
  for k from 1 to N:
    for i from 1 to N:
      for j from 1 to N:
        W[i][j] ← W[i][j] ⊕ (W[i][j] ⊗ W[k][k]* ⊗ W[k][j])
  return W
```

9. Floyd-Warshall Algorithm $O(N^3)$

```
def Floyd-Warshall(G):
  W be a N×N adjacency matrix of graph G
  d be a N×N array of minimum distance to ∞
  for each edge (u,v):
    d[u][v] ← W[u][v]
  for each vertex v:
    d[v][v] ← 0
  for k from 1 to N:
    for i from 1 to N:
      for j from 1 to N:
        if d[i][j] > d[i][k] + d[k][j]:
          d[i][j] ← d[i][k] + d[k][j]
  return d
```

Lehmann with tropical semiring is Floyd-Warshall

10. Semiring Matrix Multiplication $O(N^3)$

```
def SemiringMatrixMultiplication(A,B):
  A and B be square matrices of N×N
  C be an empty N×N matrix
  for n from 1 to N:
    for p from 1 to N:
      sum ← 0
      for m from 1 to N:
        sum ← sum ⊕ A[n][m] ⊗ B[m][p]
      C[n][p] ← sum
  return C
```

11. Floyd-Warshall Matrix Multiplication

```
def Floyd-WarshallMatrixMultiplication(A,B):
  W¹ be adjacency matrix of paths of length 1
  for each vertex k in N:
    Wᵏ = 0
  for each vertex k in N:
    Wᵏ = Wᵏ ⊕ (Wᵏ⁻¹ ⊗ W¹)
  return Wᵏ
```

XI. Axes of Modeling

1. Maximum Likelihood Estimation

$L(\theta) = -\sum_{i \leq n} \log p(y_i | x_i, \theta)$, the negative log-likelihood. The MLE minimizes $\mathbb{E}[(\theta - \theta^*)^2]$ as $n \to \infty$. Can yield the lowest KL-divergence. Fast. **Warning**: MLE can only be computed for probabilistic models, and if N is not sufficient, high variance and overfitting.

2. Parameter Estimation (MLE examples)

Gaussian: $p(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$, $LL = -N \log(\sigma) - \frac{N}{2} \log(2\pi) - \frac{1}{\sigma^2} \sum_{i=1}^{N} (x_i - \mu)^2$, $\frac{\partial LL}{\partial \mu} \to \mu = \frac{1}{N} \sum_{i=1}^{N} x_i$, $\frac{\partial LL}{\partial \sigma} \to \sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2}$. **Poisson**: $LL = -n\theta + \sum_{i=1}^{N} x_i \cdot \log(\theta) - \sum_{i=1}^{N} \log(x_i!)$, $\frac{\partial LL}{\partial \theta} \to \theta = \frac{1}{N} \sum_{i=1}^{N} x_i$.

3. (Weight) Regularization

- **Lasso**: $\mathcal{L}_{l_1}(\theta) = \mathcal{L}(\theta) + \lambda ||\theta||_1$, makes **many** coefficients to be 0. **No closed form solution**.
- **Ridge (L2)**: $\mathcal{L}_{l_2}(\theta) = \mathcal{L}(\theta) + \lambda ||\theta||_2^2$, shrinks parameters to small non-zero values. **Closed form**: $\beta = (X^T X + \lambda I)^{-1} X^T Y$.

4. Bayesian Inference and Bayes Rule

Bayesian inference involves a prior, likelihood, and posterior $p(\theta | x_1, \ldots, x_n) \propto p(x_1, \ldots, x_n | \theta) \cdot p(\theta)$. **Bayes rule**: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. *Note*: When having a strong prior, **Bayesian is preferred over MLE**.

5. Model Evaluation

Loss functions can be **directly optimized** during training; **evaluation metrics** may include **any aspect** of the model.
- **Curve scores**: AUC-ROC, AUC-PRC (precision-recall curve).
- **Confusion matrix**: precision = TP/Predicted condition positive (PCP), recall = TP/CP, accuracy = TP + TN / N, etc.
- $\boldsymbol{F_\beta}$ **score**: $F_\beta = (1 + \beta^2) \frac{\text{precision} \cdot \text{recall}}{\beta^2 \text{precision} + \text{recall}}$

6. Hypothesis Testing (example)

Given $X_1 \cdots X_n \sim N(\theta, 1)$ i.i.d., test $H_0: \theta = 0$, $H_1: \theta = 1$. $R = \left\{(x_1 \cdots x_n) \in \mathbb{R}^n : \frac{1}{n} \sum_{i \leq n} x_i > c\right\}$ rejection region. Find c to have test size of α, then $\alpha = \mathbb{P}\left(\frac{1}{n} \sum_{i \leq n} x_i \geq c \middle| H_0\right) = \mathbb{P}(\bar{x} \geq c | H_0) = \mathbb{P}(\sqrt{n}\bar{x} \geq \sqrt{n}c | H_0) = \mathbb{P}(z \geq \sqrt{n}c | H_0) = 1 - \Phi(\sqrt{n}c)$, then $c = \frac{1}{\sqrt{n}} \Phi^{-1}(1 - \alpha)$.
Power under H_1 is $\mathbb{P}\left(\frac{1}{n} \sum_{i \leq n} x_i \geq c \,|H_1\right) = \mathbb{P}(\bar{x} \geq c | H_1) = \mathbb{P}(\sqrt{n}(\bar{x} - 1) \geq \sqrt{n}(c - 1) | H_1) = \mathbb{P}(z > \sqrt{n}(c - 1) | H_0) = 1 - \Phi\left(\sqrt{n}(c - 1)\right)$.

7. P-value (example)

$H_0: \theta \in \Theta_0$, $H_1: \theta \in \Theta_1$. For every $\alpha \in (0,1)$, we have a size α test with rejection R_α. Let $x^n = \{x_1 \ldots x_n\}$ be the realization of a sample $X^n = \{X_1 \ldots X_n\}$, then p-value $= \inf\{\alpha : x^n \in R_\alpha\}$. Suppose reject H_0 iff $T(X^n) \geq c_\alpha$, then p-value $= \sup_{\theta \in \Theta_0} \mathbb{P}(T(X^n) \geq T(x^n))$, x^n is observed from X^n. If $\Theta_0 = \{\theta_0\}$, then p-value $= \mathbb{P}_{\theta_0}(T(X^n) \geq T(x^n))$.

XII. Supplemental Tips

1. Common Trig Identity and Derivatives

$\sin(0) = 0$, $\cos(0) = 1$, $\tan(0) = 0$, $\sin'(x) = \cos(x)$, $\cos'(x) = -\sin(x)$, $(\sqrt{x})' = \frac{1}{2\sqrt{x}}$, $(e^{ax})' = ae^{ax}$, $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ sigmoid, $\left(\frac{1}{x}\right)' = -\frac{1}{x^2}$.

2. CRF & Softmax

$\sum_{k=1}^{K} \left(\mathrm{score}(\boldsymbol{t}^{(k)}, \boldsymbol{w}^{(k)}) - \log \sum_{t' \in \mathcal{T}^N} \exp\left(\mathrm{score}(\boldsymbol{t}', \boldsymbol{w}^{(k)})\right)\right) = \sum_{k=1}^{K} \left(\mathrm{score}(\boldsymbol{t}^{(k)}, \boldsymbol{w}^{(k)}) - T \log \sum_{t' \in \mathcal{T}^N} \frac{\exp(\mathrm{score}(\boldsymbol{t}', \boldsymbol{w}^{(k)}))}{T}\right)$. As $T \to 0$, $\sum_{k=1}^{K} \left(\mathrm{score}(\boldsymbol{t}^{(k)}, \boldsymbol{w}^{(k)}) - \max_{t' \in \mathcal{T}^N} \mathrm{score}(\boldsymbol{t}', \boldsymbol{w}^{(k)})\right)$.
Becomes **structured perception** update rule if **minibatch size and learning rate are 1**.

3. Common Semirings

Boolean: $\langle\{0,1\}, \vee, \wedge, 0, 1\rangle$ recognition; **Viterbi**: $\langle[0,1], \max, \times, 0, 1\rangle$ prob of best derivation; **Inside**: $\langle\mathbb{R}^+ \cup \{\infty\}, +, \times, 0, 1\rangle$ prob of a string; **Real**: $\langle\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0\rangle$ shortest distance; **Tropical**: $\langle\mathbb{R}^+ \cup \{\infty\}, \min, +, \infty, 0\rangle$ shortest distance with non-negative weights; **Counting**: $\langle\mathbb{N}, +, \times, 0, 1\rangle$ number of paths.

4. Kleene Star of a Semiring

$\forall x \in A$, **(1)** $x^* = \bigoplus_{n=0}^{\infty} x^n$, **(2)** $x^* = \mathbf{1} \oplus x \otimes x^*$, **(3)** $x^* = \mathbf{1} \oplus x^* \otimes x$. *Note*: $x^0 = \mathbf{1}$.

5. Linear Indexed Grammar (LIG)

$G = \langle \mathcal{N}, S, I, \Sigma, \mathcal{R} \rangle$, where $\mathcal{N}$ is non-terminals (e.g., N, S, T); S is start non-terminal; I is finite set of indices (e.g., f, g, h); Σ is alphabet; $\mathcal{R}$ is set of production rule in one of the forms: **(1)** $N[\sigma] \to \alpha M[\sigma]\beta$, **(2)** $N[\sigma] = \alpha M[f\sigma]\beta$, **(3)** $N[f\sigma] \to \alpha M[\sigma]\beta$. For **copying** (e.g., abcabc) and **mirroring** (e.g., abccba), the indices I must be chosen from Σ. **Example**: $\{a^n b^n \# c^n d^n | n \geq 0\}$, $S[\sigma] \to aS[\sigma f]d$, $S[\sigma] \to T[\sigma]$, $T[\sigma f] \to bT[\sigma]c$, $T[] \to \#$.

6. WFSA and n-gram

$p(w_1 \ldots w_M) = \prod_{m=1}^{M} p(w_m | w_{m-1} \ldots w_{m-n+1})$ the number of states is $O(|V|^{n-1})$, where n is n-gram, $|V|$ is vocabulary size.

7. Determinant, Eigenvector, Eigenvalue

$\det\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = a(ei - fh) - b(di - fg) + c(dh - eg)$.
Given $A_{n \times n}$ a matrix and $v_{n \times 1}$ eigenvector, $Av = \lambda v$, λ is eigenvalue, a scalar.

8. Transformer and Attention Complexity

Using dot-product $\boldsymbol{\alpha}^{(t)} = \mathrm{softmax}\left(\frac{\boldsymbol{q}_t^T K}{\sqrt{h}}\right)$, with $\boldsymbol{x_t} \in \mathbb{R}^h$, we define $\boldsymbol{q_t} = W_q \boldsymbol{x_t}$, $K = W_K X$, d the context window. We have $\boldsymbol{c}^{(t)} = \boldsymbol{\alpha}^{(t)} V_t = \mathrm{softmax}\left(\frac{q_t K^T}{\sqrt{h}}\right) V_t$, $t \in [1, \cdots, n]$, $\boldsymbol{q_t} \in \mathbb{R}^{1 \times h}$, $X \in \mathbb{R}^{n \times h}$, $W_q, W_k, W_v \in \mathbb{R}^{h \times h}$, $K_t, V_t \in \mathbb{R}^{d \times h}$. Then $Q, K, V = XW_{\{q,k,v\}} \in \mathbb{R}^{n \times h} \to \boldsymbol{O(nh^2)}$, $\boldsymbol{\alpha}^{(t)} V_t \in \mathbb{R}^{1 \times h} \to \boldsymbol{O(nhd)}$, n comes from t.

9. Transformer Parameters Count

V vocabulary, E embedding, **embedding** has VE. **Positional encoding** has LE, L is sequence length. **Multi-head attention** (Q, K, V) has $3E^2$, bias $3E$, projection weight and bias $E^2 + E$, total $4E^2 + 4E$. **FFN** has $8E^2 + 5E$, where forward has $4E^2 + 4E$, projection has $4E^2 + E$. **Normalization** $4E$.

10. Sentiment Analysis

Classifying utterances according to how they make the interlocutor feel, e.g., movie review, spam detection, recommender system, etc. **(1) Embedding**: map words/tokens to vectors that encode semantic meaning (one-hot, skip-gram, BERT, ELMo). **(2) Pooling**: aggregate token vectors into a fixed-size representation for classification (mean, max, sum pooling). **(3) Backprop. (4) Softmax.** *Note*: Skip-gram $p(c|w) = \frac{1}{Z(w)} \exp\{\mathrm{e_{wrd}}(w) \cdot \mathrm{e_{ctx}}(w)\}$, two outputs - $\{\mathrm{e_{wrd}}(w)\}_{w \in V}$ and $\{\mathrm{e_{ctx}}(w)\}_{w \in V}$, V is the set of word types in corpus, $\mathrm{e}(w) \in \mathbb{R}^d$, $O(kC)$.