

L101: Introduction to Structured Prediction


Ryan Cotterell

What is structured prediction?

- It's just multi-class classification!
- Definition: **Structured**
 - Something in the problem is exponentially large
- Definition: **Structured Prediction:**
 - The output space of the prediction problem is exponentially large

Recall Logistic Regression

*We will define
this later*



- **Goal:** to construct a probability distribution

$$p(y \mid x) = \frac{\exp\{score(y, x)\}}{\sum_{y' \in \mathcal{Y}} \exp\{score(y', x)\}}$$

- The major question: What if $|\mathcal{Y}|$ is really, really big?
- Can we find an efficient algorithm for computing that sum?

Structured Prediction in a Meme

$$|\mathcal{Y}| = 2$$

$$|\mathcal{Y}| = n$$

$$|\mathcal{Y}| = 2^n$$



Sentiment Analysis:

Is sentiment positive or negative?

Movie Genre Prediction:

Which genre is this script?

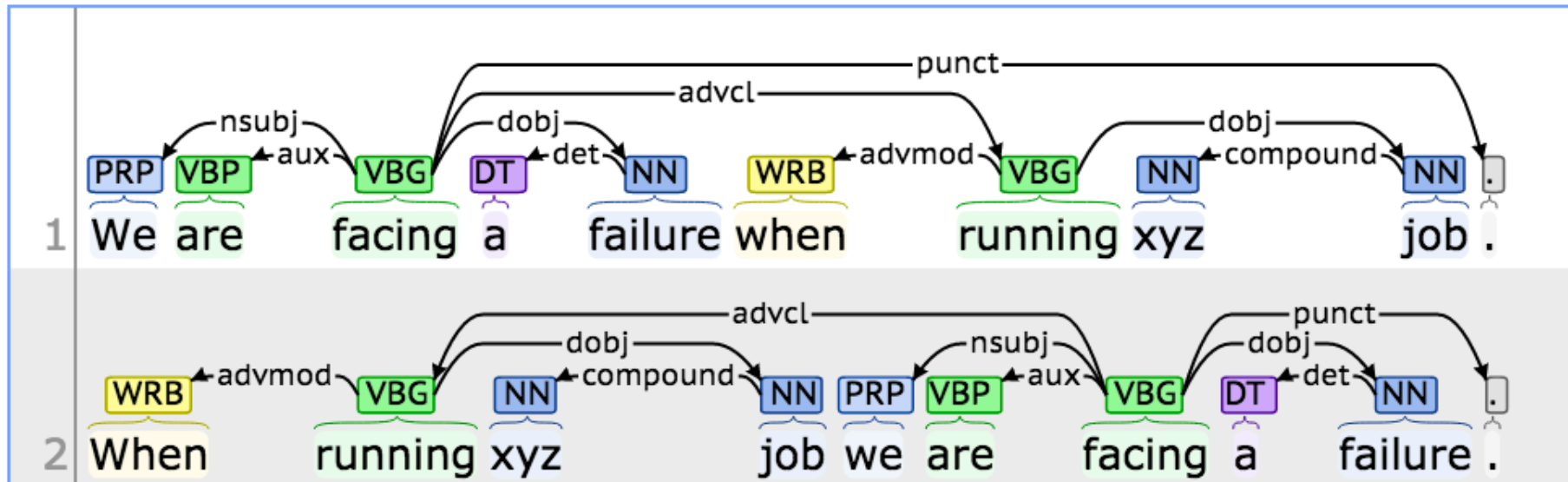
Part-of-Speech Tagging:

This sentence has which part-of-speech-tag sequence?

Predict Trees!

- Predict dependency parses from raw text
- Classic problem in NLP

Basic Dependencies:



Predict Subsets!

- Determinantal Point Processes
 - A distribution over subsets

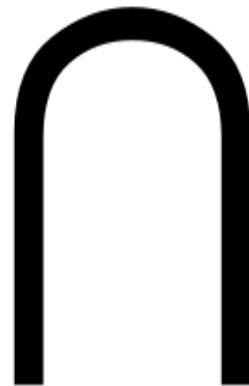
$$\begin{array}{c} \mathcal{Y} \\ \bullet \bullet \bullet \bullet \\ 1 \ 2 \ 3 \ 4 \end{array} \quad K = \begin{array}{|c|c|c|c|} \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \end{array} \quad A = \{1, 3\}$$

$$\mathcal{P}(A \subseteq \mathbf{Y}) = \mathcal{P}(\bullet \text{?} \bullet \text{?}) = \left| \begin{array}{|c|c|c|c|} \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \end{array} \right| = \left| \begin{array}{|c|c|} \hline \square & \square \\ \hline \square & \square \\ \hline \end{array} \right|$$

Why isn't Structured Prediction just Statistics?

- Computer scientists develop combinatorial algorithms professionally
 - Minimum spanning tree, shortest path problems, maximum flow, LP relaxations
- Structured prediction is the intersection of algorithms and high-dimensional statistics

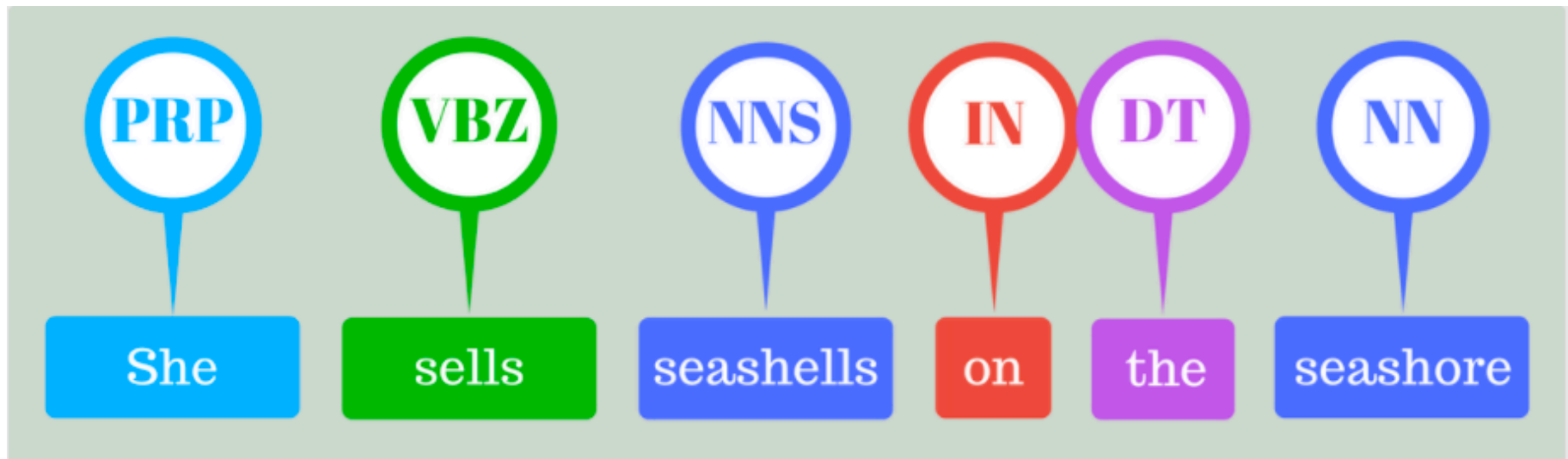
Statistics



**(Theoretical)
Computer
Science**

Deep Dive into Discriminative Tagging

- Assign each word in a sentence a coarse-grained grammatical category
 - Noun, Verb, Adjective, Adverb, Determiner, etc...
- Arguably, the simplest structured prediction problem in NLP



Back in 2001...



Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data

John Lafferty^{†*}
Andrew McCallum^{*†}
Fernando Pereira^{*‡}

LAFFERTY@CS.CMU.EDU
MCCALLUM@WHIZBANG.COM
FPEREIRA@WHIZBANG.COM

*WhizBang! Labs—Research, 4616 Henry Street, Pittsburgh, PA 15213 USA

[†]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA

[‡]Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA

Abstract

We present *conditional random fields*, a framework for building probabilistic models to segment and label sequence data. Conditional random fields offer several advantages over hidden Markov models and stochastic grammars for such tasks, including the ability to relax strong independence assumptions made in those models. Conditional random fields also avoid a fundamental limitation of maximum entropy Markov models (MEMMs) and other discriminative Markov models based on directed graphical models, which can be biased towards states with few successor states. We present iterative parameter estimation algorithms for conditional

random fields that maximize the joint likelihood of training examples. To define a joint probability over observation and label sequences, a generative model needs to enumerate all possible observation sequences, typically requiring a representation in which observations are task-appropriate atomic entities, such as words or nucleotides. In particular, it is not practical to represent multiple interacting features or long-range dependencies of the observations, since the inference problem for such models is intractable.

This difficulty is one of the main motivations for looking at conditional models as an alternative. A conditional model specifies the probabilities of possible label sequences given an observation sequence. Therefore, it does not expend modeling effort on the observations, which at test time are fixed anyway. Furthermore, the conditional probabil-

What is a *score* function for tagging?

- Arbitrary function that takes a word sequence and a tag as input and tell you how good they are together

$$\textit{score}(\mathbf{w}, \mathbf{t}) = \textit{“goodness”}(\mathbf{w}, \mathbf{t})$$

$$p(\mathbf{t} \mid \mathbf{w}) \propto \exp\{\textit{score}(\mathbf{w}, \mathbf{t})\}$$

You *score* function can be any function!

Linear Function

(dot product of a weight vector
and a feature function)

$$\textit{score}(\mathbf{w}, \mathbf{t}) = \boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{w}, \mathbf{t})$$

$$\textit{score}(\mathbf{w}, \mathbf{t}) = \textit{Neural-Network}(\mathbf{w}, \mathbf{t})$$

Non-linear Function

(neural network)

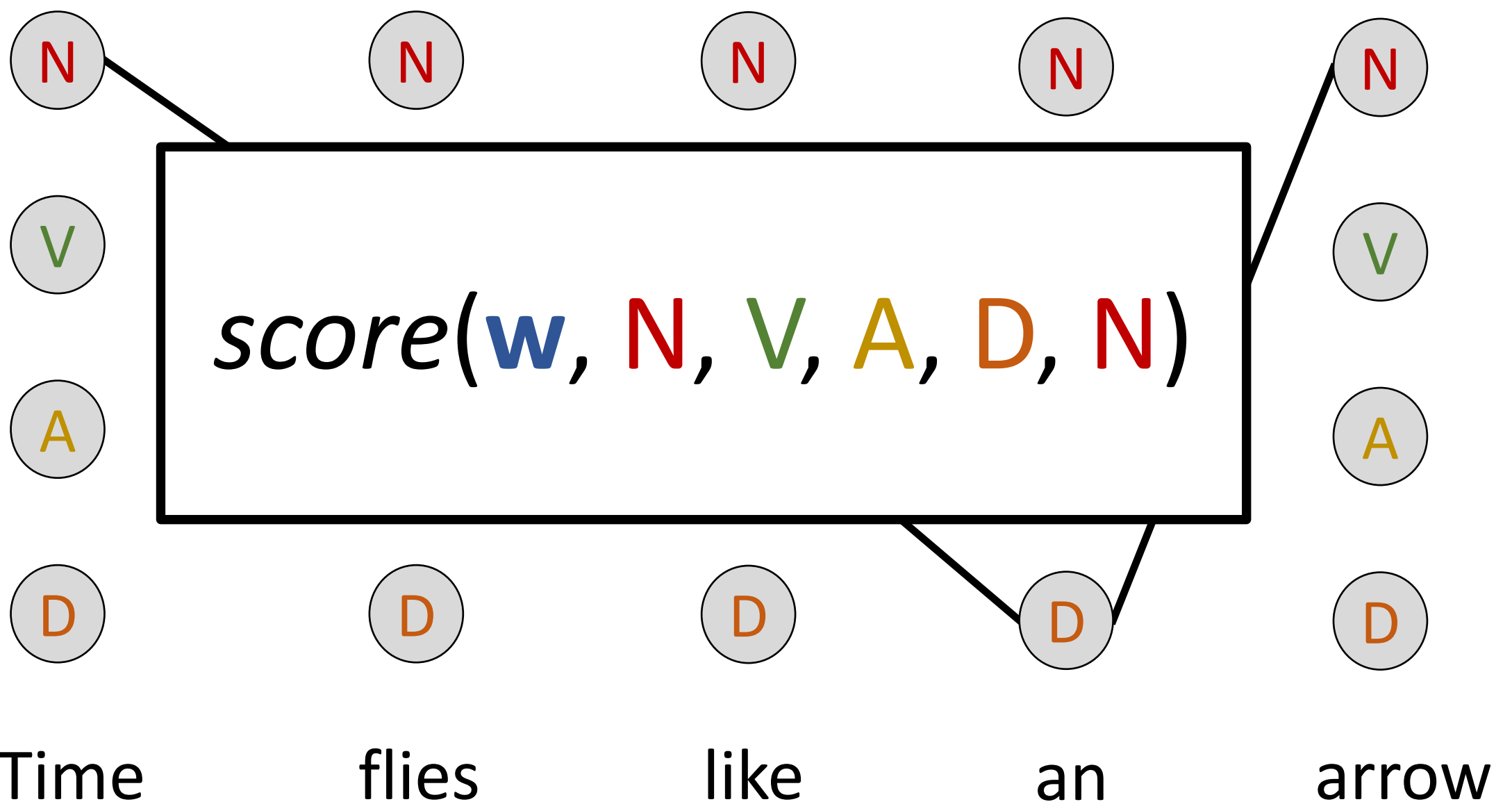
Noun

Verb

Adv

Det

w



How do we normalize?

- Why is this hard?
 - There are an exponential number of summands

$$p(\mathbf{t} \mid \mathbf{w}) = \frac{\exp\{score(\mathbf{t}, \mathbf{w})\}}{\sum_{\mathbf{t}' \in \mathcal{T}^n} \exp\{score(\mathbf{t}', \mathbf{w})\}} \mathcal{O}(|\mathcal{T}|^n)$$

- \mathcal{T} is the set of tags (typically about 12)
- $|\mathcal{T}^n| = |\mathcal{T}|^n$
- Naïve algorithm runs in $\mathcal{O}(|\mathcal{T}|^n)$
- The normalizer is terms the partition function (Zustandssumme)

$$Z(\mathbf{w}) = \sum_{\mathbf{t}' \in \mathcal{T}^n} \exp\{score(\mathbf{t}', \mathbf{w})\}$$

What if we assume structure?

- Does the problem get any easier?
- Yes! And we'll see how
- Structured prediction is about exploiting combinatorial structure for great good

Example of Structure: Additively Decomposable Score Functions

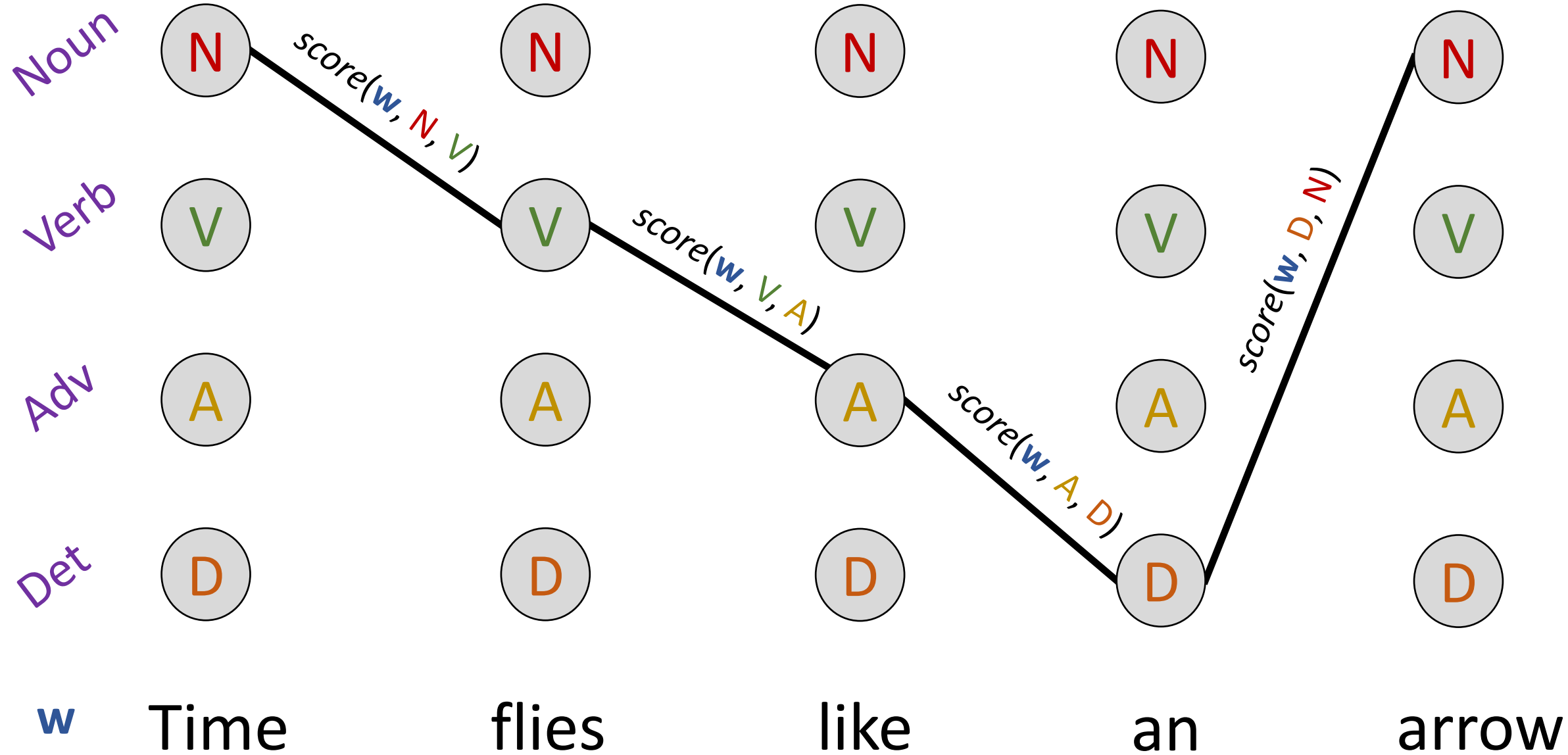
tag sequence of length n

t_0 is start-of-tagging symbol

$$score(\mathbf{w}, \mathbf{t}) = \sum_{i=1}^n score(\mathbf{w}, t_{i-1}, t_i)$$

word sequence
(sentence) of length n

tag bigram



Additive Decomposable Score Functions

$$\textit{score}(\mathbf{w}, \mathbf{N}, \mathbf{V}, \mathbf{A}, \mathbf{D}, \mathbf{N}) =$$

$$\textit{score}(\mathbf{w}, \mathbf{N}, \mathbf{V}) + \textit{score}(\mathbf{w}, \mathbf{V}, \mathbf{A}) + \textit{score}(\mathbf{w}, \mathbf{A}, \mathbf{D}) + \textit{score}(\mathbf{w}, \mathbf{D}, \mathbf{N})$$

$$p(\mathbf{t} \mid \mathbf{w}) \propto \exp \{ \textit{score}(\mathbf{w}, \mathbf{t}) \}$$

Probability Distribution:

$$= \exp \left\{ \sum_{i=1}^n \textit{score}(\mathbf{w}, t_{i-1}, t_i) \right\}$$

$$= \prod_{i=1}^n \exp \{ \textit{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

How do we Compute Z?

$$Z(\mathbf{w}) = \sum_{\mathbf{t}_1^n \in \mathcal{T}^n} \prod_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \sum_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \sum_{t_n \in \mathcal{T}} \prod_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \sum_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \prod_{i=1}^{n-1} \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \times \sum_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

$$= \sum_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \times \sum_{t_2 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_1, t_2) \} \times \cdots \times \sum_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

$$= \sum_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \times \beta(\mathbf{w}, t_1)$$

$$= \beta(\mathbf{w}, t_0)$$

A Simple Dynamic Program

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

$$\beta(\mathbf{w}, t_i) = \sum_{t_{i+1} \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_i, t_{i+1}) \} \times \beta(\mathbf{w}, t_{i+1})$$

- You've seen this before! It's just the backward algorithm from HMMs
- Same algorithm, because the lattice structure is the same
- Runtime complexity? Space complexity?

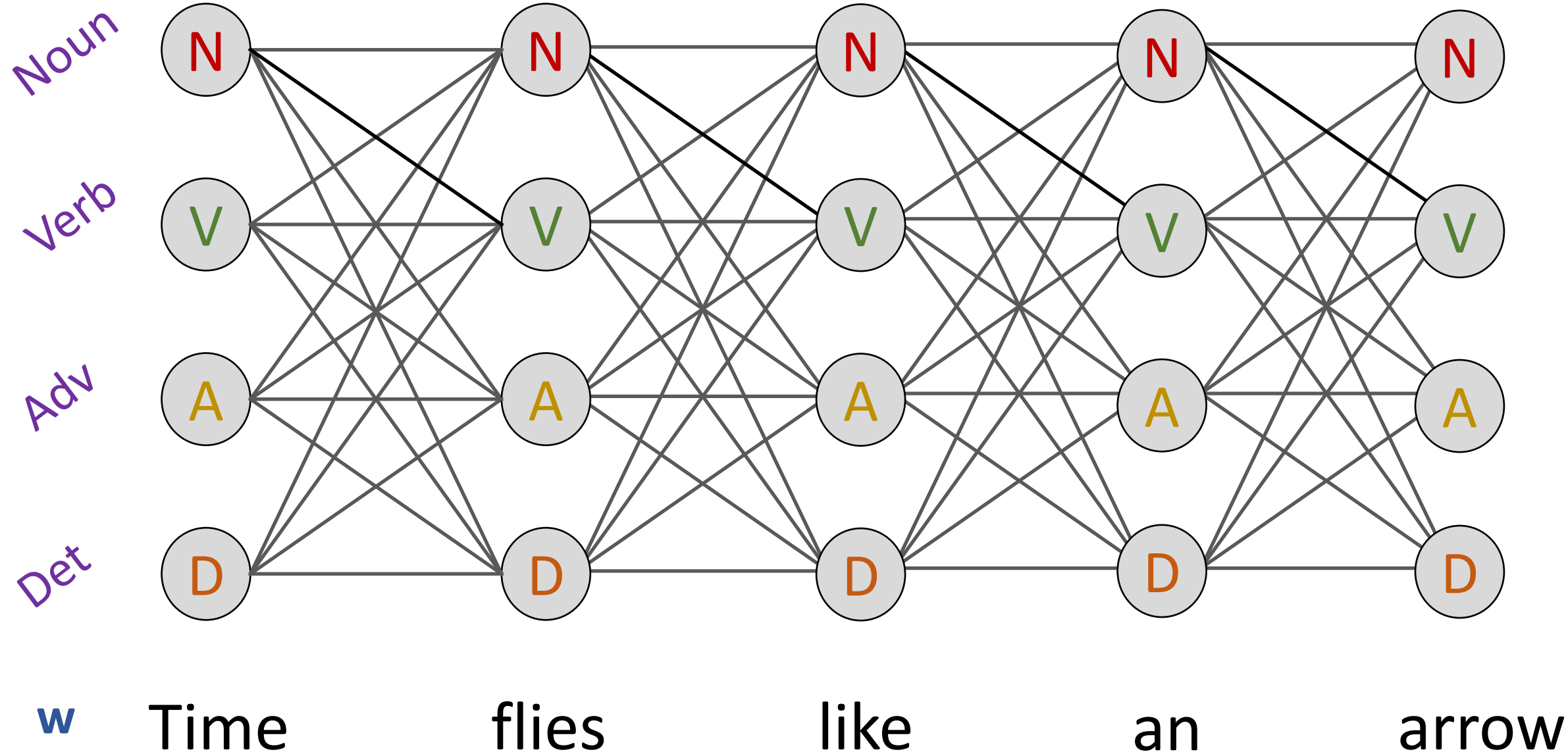
Graphical Representation

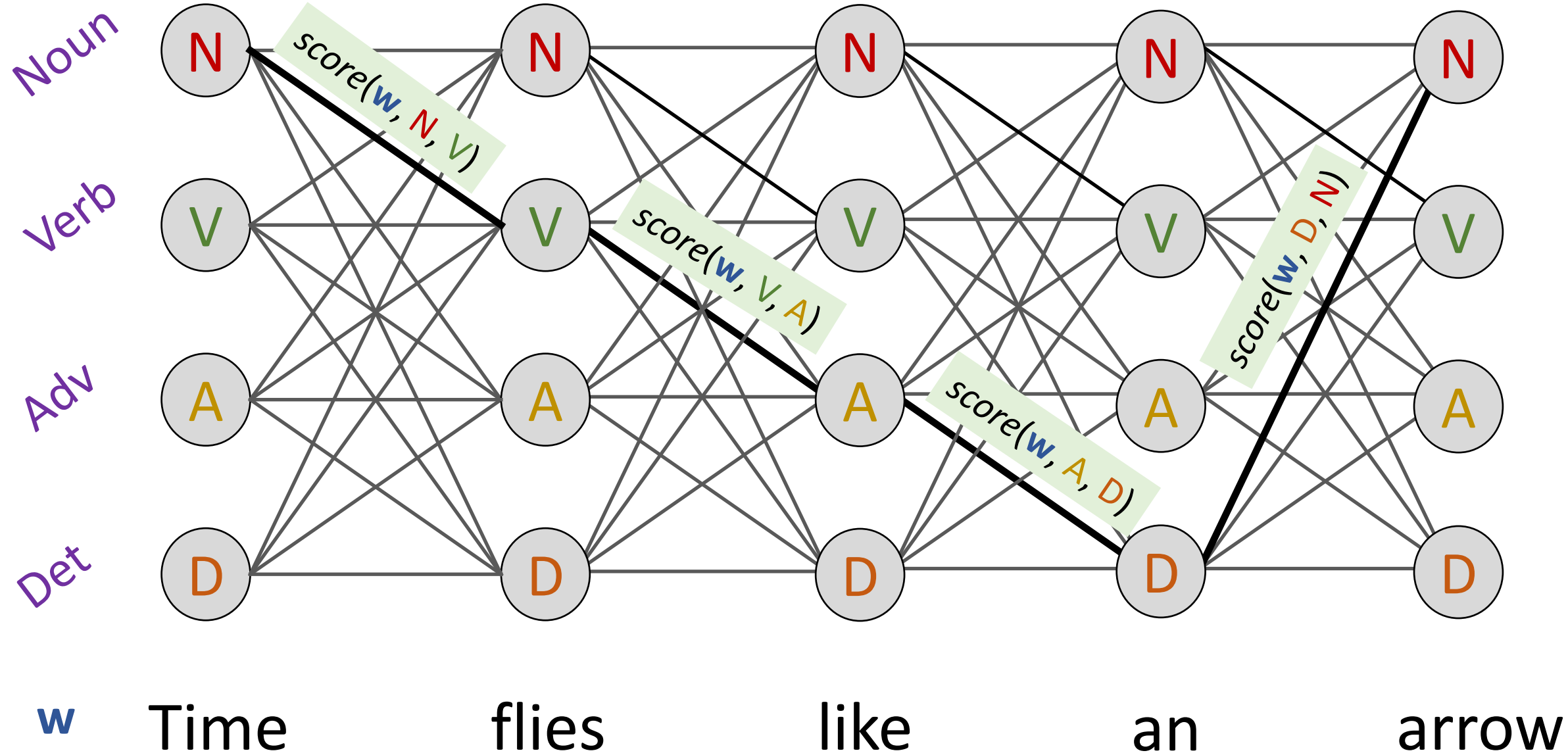
- That was a lot of algebra, huh!
- To gain insight insight into this problem, let's consider a related problem of finding the highest-scoring tagging
 - Note that multiplication distributes over max (for non-negative values) just like sum does, so the same derivation holds!

$$\gamma(\mathbf{w}, t_n) = 1$$

$$\gamma(\mathbf{w}, t_i) = \max_{t_{i+1} \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_i, t_{i+1}) \} \times \gamma(\mathbf{w}, t_{i+1})$$

Every arc has a weight score(w , t , t')





Shortest Path in a Graph

- Find the shortest path in a (weighted) graph
 - A very common problem in computer science
 - In every computer science textbook
 - We just reduced (MAP) inference to a problem you already knew!



Dijkstra



Bellman



Ford



Viterbi

One Problem, Many Algorithms

- Tagging lattices are acyclic, so we don't have to worry about special solutions for cycles
 - Dealing with cycles is a focal point in algorithms classes
- Dijkstra's can incorporate a heuristic to make it faster
 - This is known as A* search

What does “short” mean anyway?

- You just need to redefine “short” to be a bit more abstract
- The computation of the partition function Z is a shortest-path problem in the abstract
- Can we go more general?
 - We will parameterize the dynamic program by a semiring
 - This gives us a family of algorithms

Introduction to Semirings

- A semiring is an algebraic structure
- The above is tedious! So, let's come up with a better intuition

Definition 2. A *semiring* is a 5-tuple $R = (A, \oplus, \otimes, \bar{0}, \bar{1})$ such that

1. $(A, \oplus, \bar{0})$ is a commutative monoid.
2. $(A, \otimes, \bar{1})$ is a monoid.
3. \otimes distributes over \oplus : for all 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. $\bar{0}$ is an *annihilator* for \otimes : for all a in A , $\bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$.

A Semiring is object where this Derivation Holds

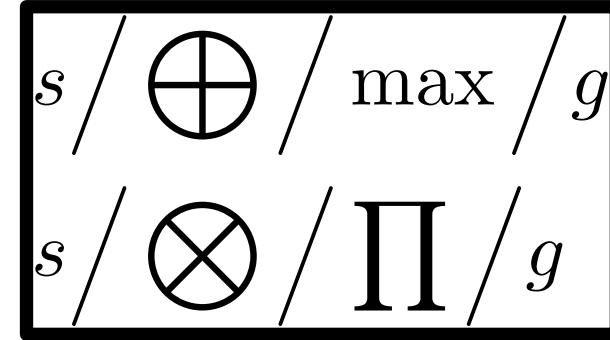
$$= \bigoplus_{\mathbf{t}_1^n \in \mathcal{T}^n} \bigotimes_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \bigoplus_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \bigoplus_{t_n \in \mathcal{T}} \bigotimes_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \bigoplus_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \bigotimes_{i=1}^{n-1} \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \otimes \bigoplus_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

$$= \bigoplus_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \otimes \bigoplus_{t_2 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_1, t_2) \} \otimes \cdots \otimes \bigoplus_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

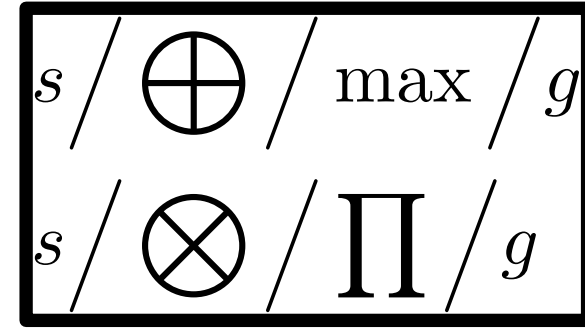
$$\beta(\mathbf{w}, t_0) = \bigoplus_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \otimes \beta(\mathbf{w}, t_1)$$



Backward Algorithm

$$\begin{aligned} Z(\mathbf{w}) &= \sum_{\mathbf{t}_1^n \in \mathcal{T}^n} \prod_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \\ &= \sum_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \sum_{t_n \in \mathcal{T}} \prod_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \\ &= \sum_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \prod_{i=1}^{n-1} \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \times \sum_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \} \\ &= \sum_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \times \sum_{t_2 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_1, t_2) \} \times \cdots \times \sum_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \} \\ &= \sum_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \times \beta(\mathbf{w}, t_1) \\ &= \beta(\mathbf{w}, t_0) \end{aligned}$$

Viterbi Algorithm



$$= \bigoplus_{\mathbf{t}_1^n \in \mathcal{T}^n} \bigotimes_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \bigoplus_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \bigoplus_{t_n \in \mathcal{T}} \bigotimes_{i=1}^n \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \}$$

$$= \bigoplus_{\mathbf{t}_1^{n-1} \in \mathcal{T}^{n-1}} \bigotimes_{i=1}^{n-1} \exp \{ \text{score}(\mathbf{w}, t_{i-1}, t_i) \} \otimes \bigoplus_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

$$= \bigoplus_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \otimes \bigoplus_{t_2 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_1, t_2) \} \otimes \cdots \otimes \bigoplus_{t_n \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_{n-1}, t_n) \}$$

$$\beta(\mathbf{w}, t_0) = \bigoplus_{t_1 \in \mathcal{T}} \exp \{ \text{score}(\mathbf{w}, t_0, t_1) \} \otimes \beta(\mathbf{w}, t_1)$$

Examples of Semiring

Semiring	Set	\oplus	\otimes	$\bar{0}$	$\bar{1}$	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

Table 2: Examples of semirings

Conclusion

- Deep-dive into discriminative tagging with CRFs
- Generalized the idea of a shortest-path problem to semirings
- You already knew the content of this lecture from your algorithms class!

Fin