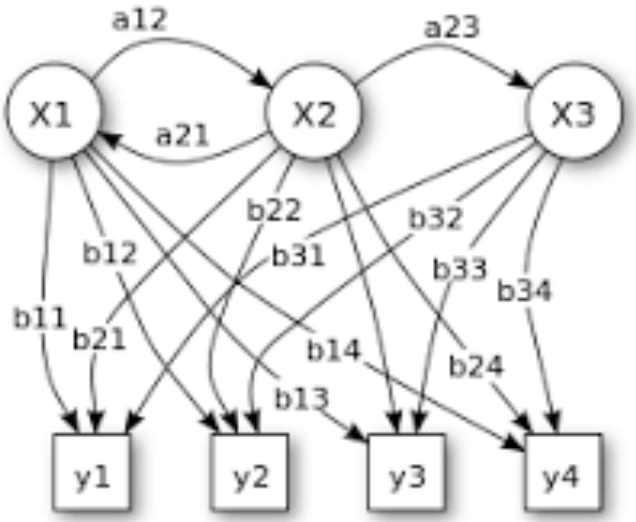
Introduction to Markov Models

Kasthuri Kannan, PhD
Assistant Professor of Pathology
New York University

Overview of Topics

- Introduction to Markov Processes
- Hidden Markov Models
- Forward Algorithm
- Viterbi Algorithm
- Tutorial

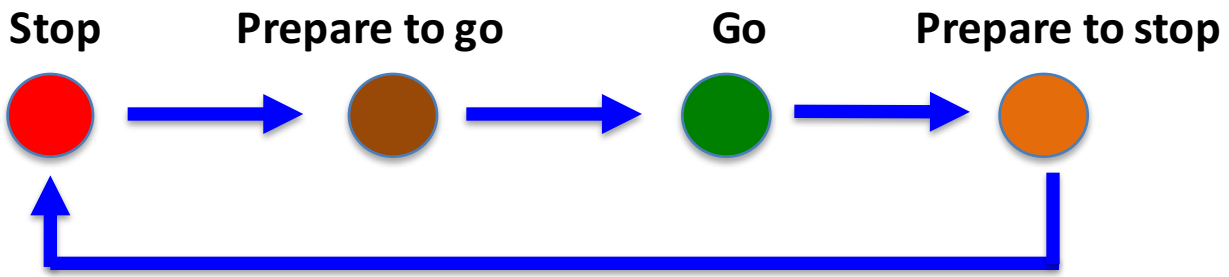


Introduction to Markov Processes

- Interested in finding patterns appearing over time
  - Sequence of events
- Appear in many areas in nature
  - Biological sequences (DNA, RNA, Proteins etc.)
  - Sequences of words, natural language processing
  - Weather phenomenon etc.
- Interested in knowing if the sequences makes useful patterns which can be modeled

Introduction to Markov Processes

- Sequence of traffic lights – red - red/amber – green – amber – red.
  - Can be viewed as a trellis diagram



- Each state is dependent solely on previous state (deterministic system) – observable Markov process

Introduction to Markov Processes

- Deducing weather from a piece of seaweed
- Soggy → Sun
- Damp means can't be sure

- Note: State of the weather is not dependent on the state of the seaweed so we can say something (like raining)
- Another clue would be the state in the preceding day
  - by combining this knowledge we will be able to come to a better forecast

Introduction to Markov Processes

- Consider systems which generate probabilistic patterns in time – such as weather fluctuating between sunny and rainy, or signal going from amber to red

- We the look at systems where what we wish to predict is not what we observe
  - the underlying hidden system

- In seaweed example, the hidden system would be the actual weather

- In traffic light example, the hidden system could be the actual signal while observed sequence would be traffic (cars, busses) stopping or moving

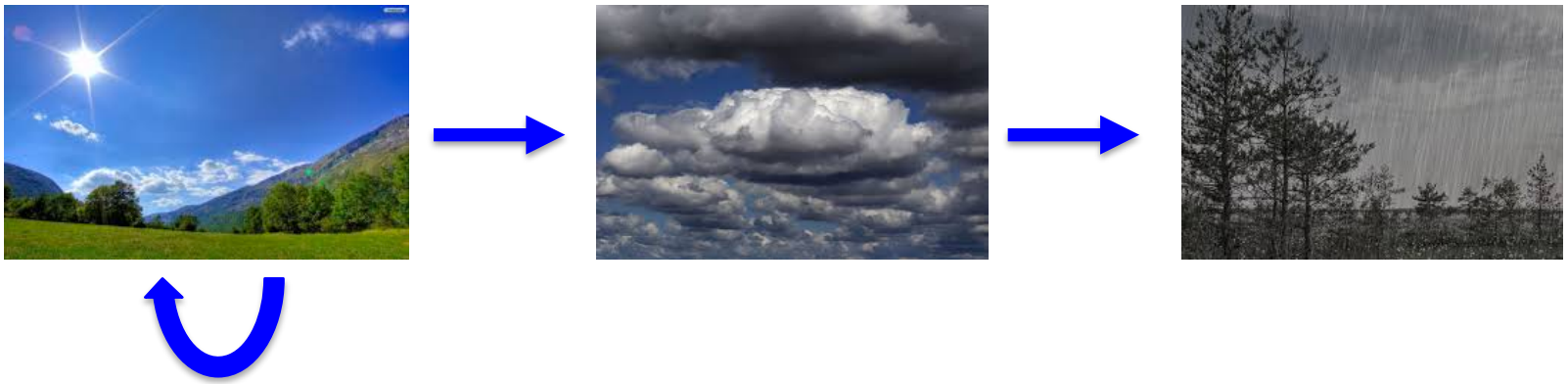
Introduction to Markov Processes

- Some problems can be solved once the system is modeled
- What the weather was for a week given each day's seaweed observation
- Given a sequence of seaweed observations, is it winter or summer?
  - Intuitively if the seaweed has been dry for a while it may be summer, else, otherwise

Introduction to Markov Processes

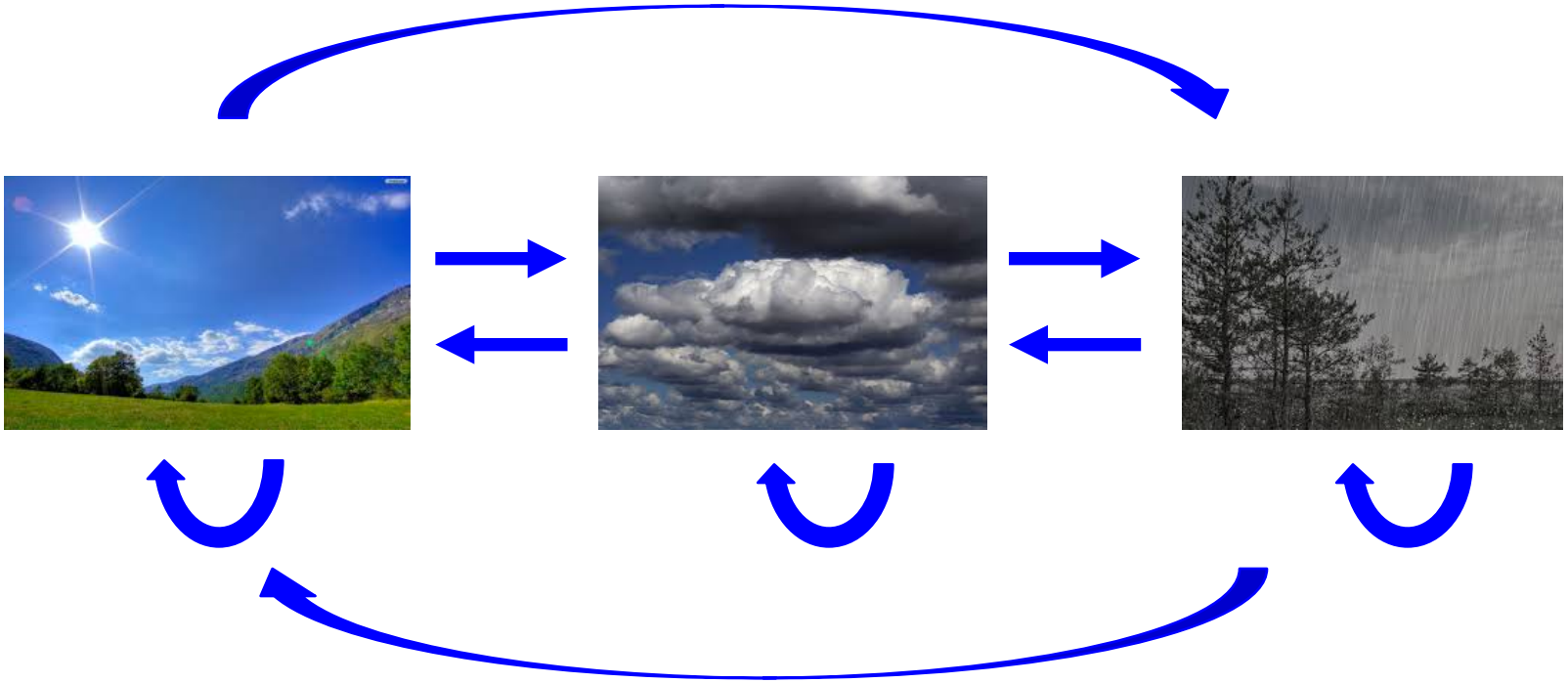
- Typically we build trellis diagram – each state depending only on the previous state

- Weather example: each sequence taking a single day (with self-transition) – one possible sequence



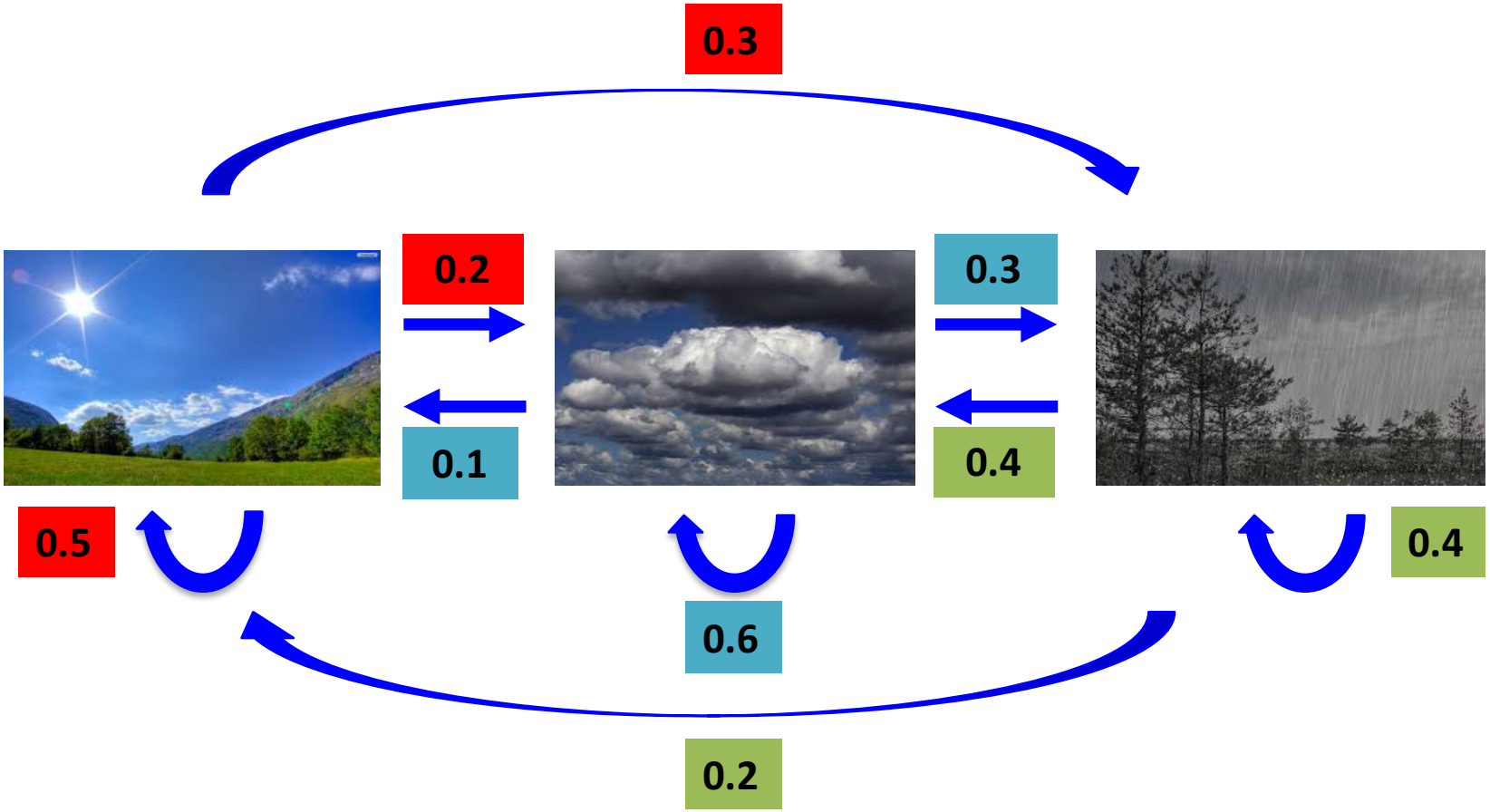
Introduction to Markov Processes

All possible sequences



Introduction to Markov Processes

All possible sequences with probabilities



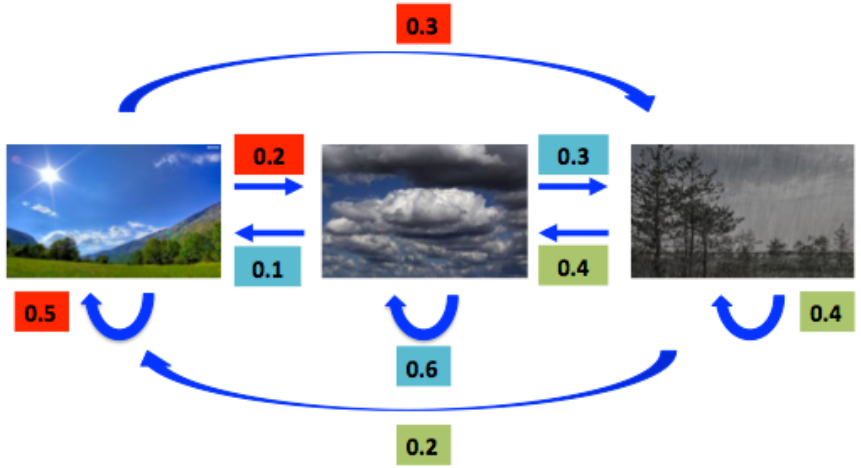
Note: Total of these colored probabilities equals 1

Introduction to Markov Processes

All possible sequences with probabilities

	Sunny	Cloudy	Rainy
Sunny	0.5	0.2	0.3
Cloudy	0.1	0.6	0.3
Rainy	0.2	0.4	0.4

Transition Matrix



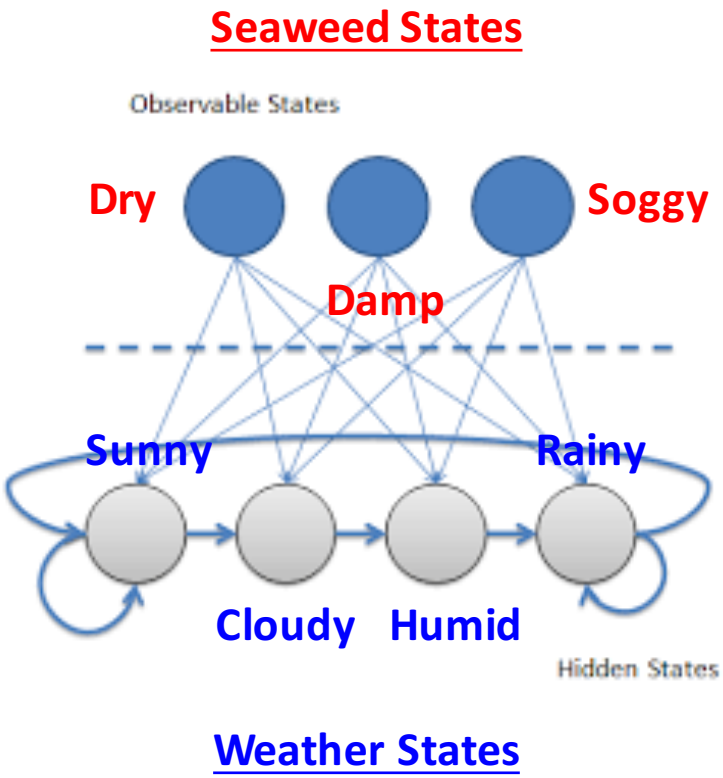
Note: Total of these colored probabilities equals 1

Introduction to Markov Processes

- Many cases the patterns are not apparent
- A crab may have access only to the seaweed and not the weather – although weather and seaweed states are closely linked
- Two set of states
  - Observable states (seaweed)
  - Hidden states (the state of the weather)
- Devise an algorithm for the crab to forecast weather from the seaweed and Markov assumption **without** actually ever seeing the weather

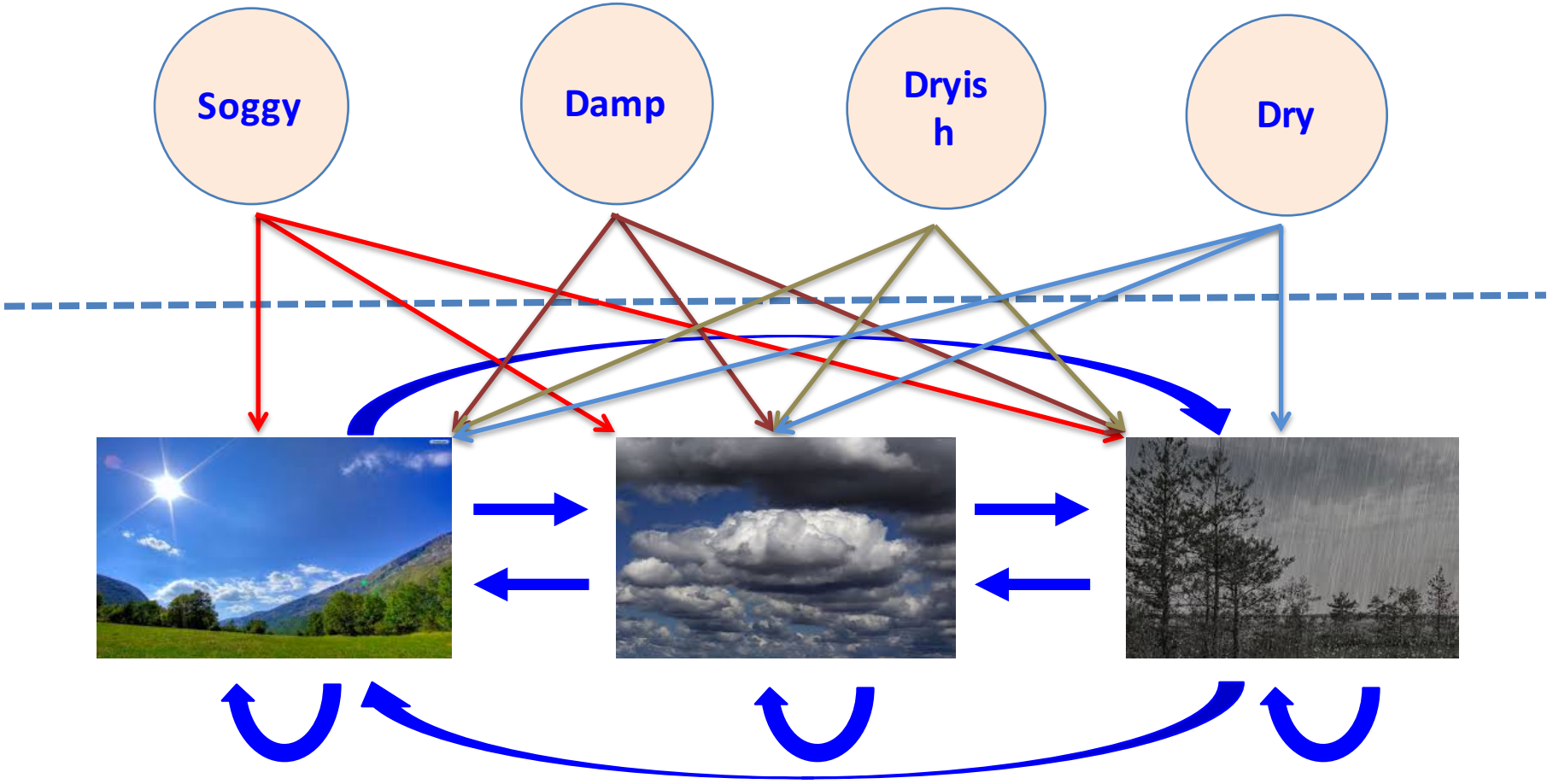
Hidden Markov Models

- Observed sequence of states is probabilistically related to hidden states
- Hidden Markov Models – modeling such processes where set of observable states are somehow related to hidden states
- – with Markov assumption
- Number of hidden states may be different from number of observable states



Hidden Markov Models

Observable States

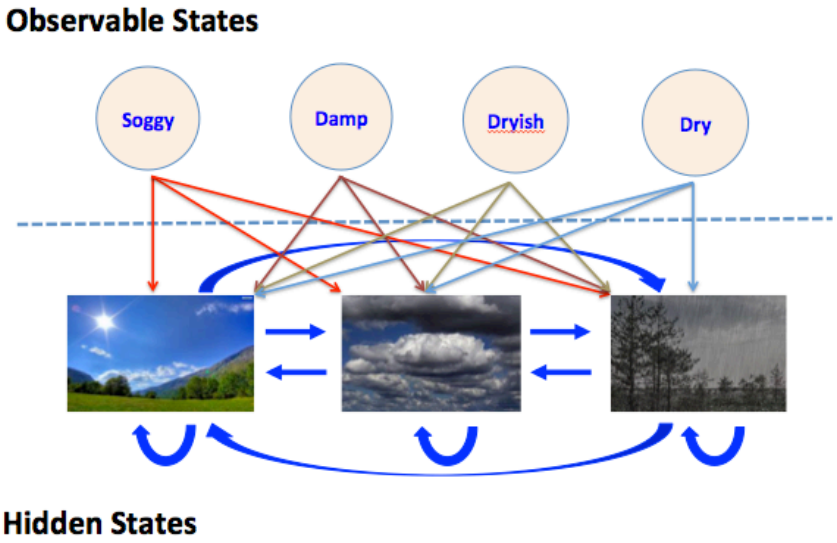


Hidden States

Hidden Markov Models

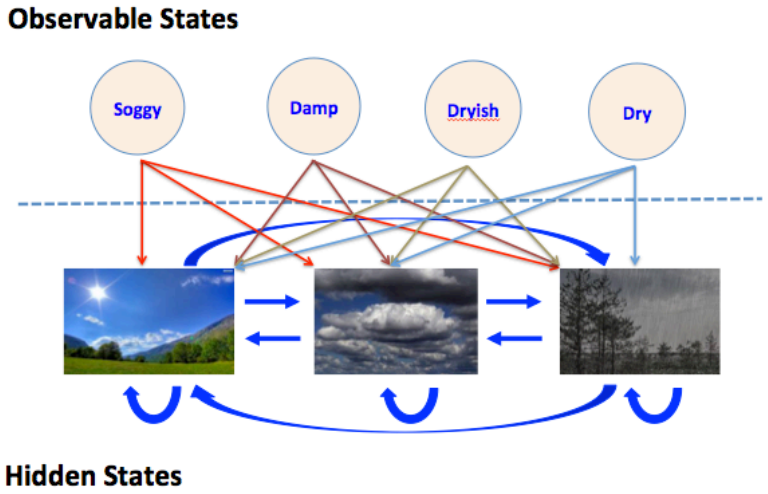
- Hidden states are modeled by simple first order Markov process

- The connections between hidden states and observable states represent the probability of generating a particular observed state given that the Markov process is in a particular hidden state



- Probabilities "entering" an observable state will sum up to 1, since it would be the sum of Pr(Obs|Sun), Pr(Obs|Cloud) and Pr(Obs|Rain)

Hidden Markov Models



Apart from the transition matrix we also have emission matrix which contains the probabilities of the observable states given a particular hidden state.
Note: sum of each matrix row is 1

Probabilities in transition and emission matrices are time independent – i.e., they don't change over time

One of the **most unrealistic assumptions** of Markov models when applied to real process.

Seaweed State Today

Weather Yesterday	Dry	Dryish	Damp	Soggy
Sunny	0.6	0.2	0.15	0.05
Cloudy	0.25	0.25	0.25	0.25
Rainy	0.05	0.10	0.35	0.5

Emission matrix/probabilities

Hidden Markov Models

- Evaluation (pattern recognition) – Finding the probability of an observed sequence given a HMM

- Decoding (pattern recognition) – Finding the sequence of hidden states that most probably generated an observed sequence

- Learning – Generating a HMM given a sequence of observations

Hidden Markov Models - Evaluation

- Number of HMMs describing different systems and sequence of observations
- Which HMM most probably generated the given sequence
  - Example, a “summer” and “winter” model for seaweed behavior to determine season
- Usually carried out by forward algorithm to calculate the probability of an observation sequence given a HMM and choose the most probable HMM

Hidden Markov Models - Decoding

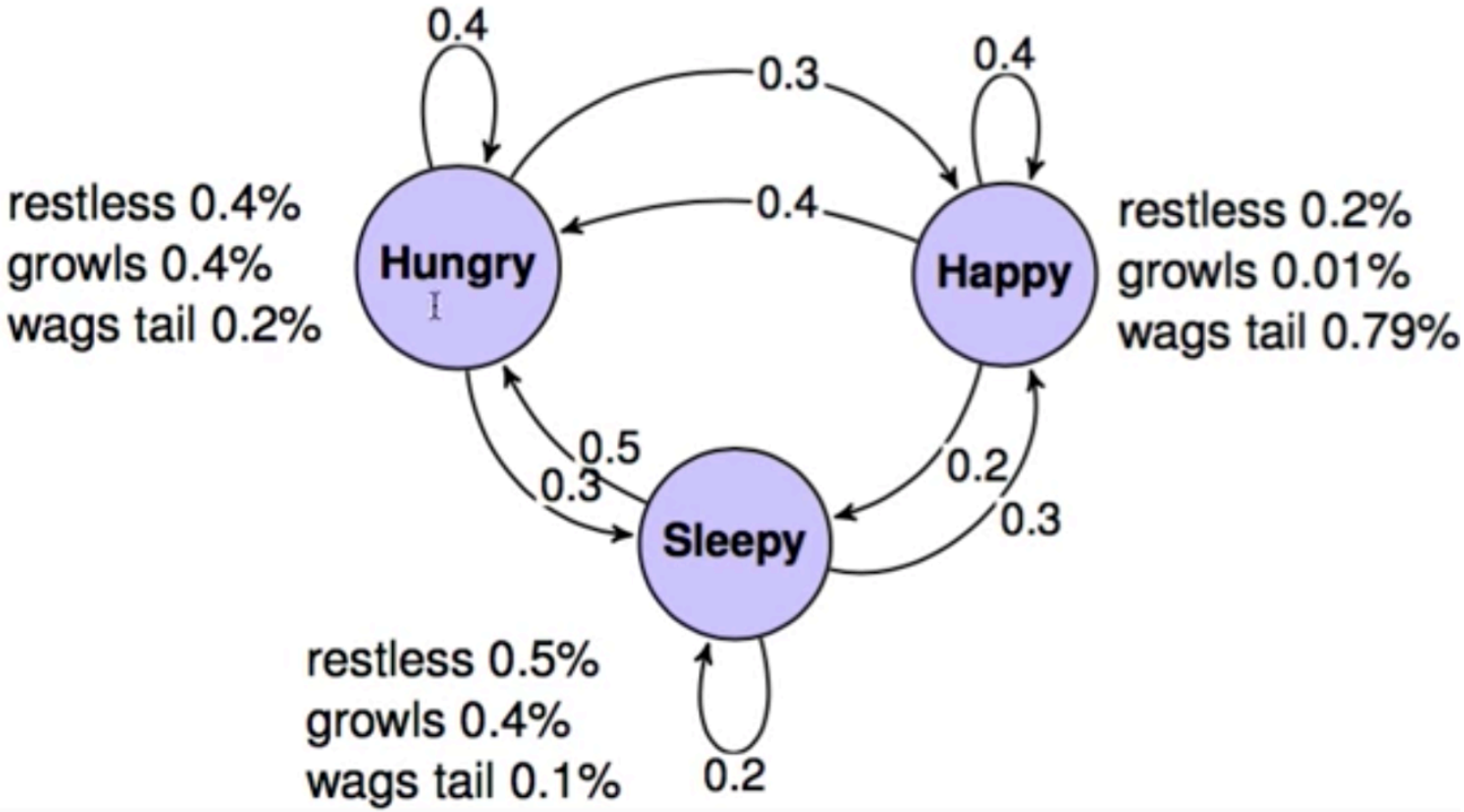
- Find the most probable sequence of hidden states given some observations
- In several instances we are interested in identifying the hidden states since they represent something of value
  - For example, chromatin states, CpG islands etc.
- In our own example, identifying the weather only knowing the status of the seaweed
- Viterbi's algorithm is handy for this process.

Hidden Markov Models - Learning

- Involves generating a HMM from a sequence of observations
- Take a sequence of observations (from a known set), known to represent a set of hidden states and fit the most probable HMM
- Baum-Welch expectation maximization (EM) can be used to identify local optimal parameters for the HMM for the Learning problem

The Forward Algorithm

A real world example



The dog's behavior depends on its state ($P(o^k|x^j)$) and its state at time t depends on its previous state at time $t-1$. ($P(x_t^j|x_{t-1}^k)$)

The Forward Algorithm

Predicting the state after n observations

If I observe the dog wagging its tail at time t and I know the dog was hungry at time $t-1$, what state is the dog in?

$o_t^k = wagTail;\ x_{t-1}^j = hungry;\ x_t^i = ?$

I need to compute:

$$\underset{x^i \in \{hungry, happy, sleepy\}}{\text{argmax}} \quad p(wagTail_t | x_t^i, hungry_{t-1})$$

$$= \underset{x^i}{\text{argmax}}\, p(o_t^k = wt | x_t^i, x_{t-1}^j = hungry)$$

using Bayes...

$$= \underset{x^i}{\text{argmax}}[p(x_t^i | x_{t-1}^j = hungry) \times p(o_t^k = wt | x_t^i)]$$

possible **state changes** x_t^i: Hungry, Happy or Sleepy?

replace x^i and determine the argmax according to the tables.

The Forward Algorithm

Predicting the state after n observations

So, we know how to compute the probability of the current observation given the previous and current state.

But at some point I need probabilities for my starting state

Solution: I can say that from my experience, dogs are hungry 30% of the time, happy 40% of the time and sleepy 30% of the time. These are the **initial** probabilities.

The Forward Algorithm

Predicting the state after n observations

Remember conditioning: $P(A) = \sum_b P(A|B)P(B)$

So, the probability of my model producing a set of observations O from a set of states $\mathcal{X}$:

$$P(O) = P(\mathcal{X}, O) = \sum_{r=1}^{R} P(O|\mathcal{X}_r)P(\mathcal{X}_r)$$

Where $\mathcal{X}_r$ is a combination of states[3] corresponding to the observations in O.

The Forward Algorithm

Predicting the state after n observations

$$P(\mathcal{X}, O) = \sum_{r=1}^{R} P(O|\mathcal{X}_r)P(\mathcal{X}_r)$$

but we know that:

$$P(\mathcal{X}_r) = \prod_{t=1}^{T} P(x_t^j|x_{t-1}^i) = \prod_{t=1}^{T} a^{ij}$$

and

$$P(O|X_r) = \prod_{t=1}^{T} P(o_t^k|x_t^j) = \prod_{t=1}^{T} e^{jk}$$

Therefore,

$$P(X, O) = \sum_{r=1}^{R} \prod_{t=1}^{T} a^{ij} e^{jk}$$

The probability of my model with R possible combinations and T time-slices is:

$$P(O) = \sum_{r=1}^{R} \prod_{t=1}^{T} a^{ij} e^{jk}$$

For N hidden states in T time slices, computation is:[4]

$$O(N^T T)!!!!!$$

The Forward Algorithm

Building computations one at a time

Let's say that

$$\alpha_i(t) = p(x_t^i, o_{1:t})$$

the probability of being in state x^i at timeslice t with a history of observations $o_{1:t}$*

$$\alpha_i(t) = p(x_t^i, o_{1:t}) = \sum_j p(x_t^i, x_{t-1}^j, o_{1:t})$$

Using the chain rule:

$$\alpha_i(t) = \sum_j p(o_t | x_t^i, x_{t-1}^j, o_{1:t-1}) p(x_t^i | x_{t-1}^j, o_{1:t-1}) p(x_{t-1}^j, o_{1:t-1})$$

The Forward Algorithm

Building computations one at a time

$$\alpha_i(t) = \sum_j \underbrace{p(o_t|x_t^i, x_{t-1}^j, o_{1:t-1})}_{p(o_t|x_t^i)} \underbrace{p(x_t^i|x_{t-1}^j, o_{1:t-1})}_{p(x_t^i|x_{t-1}^j)} \underbrace{p(x_{t-1}^j, o_{1:t-1})}_{\alpha_j(t-1)}$$

Therefore

$$\alpha_i(t) = p(o_t|x_t^i) \sum_j p(x_t^i|x_{t-1}^j)\alpha_j(t-1)$$

With

$$\alpha_i(0) = \textit{initial probability for } x^i$$

The computation time is $O(N^2 T)$ and can be represented in a **trellis**

The Forward Algorithm

Trellis for the states of the dog given observations

Suppose I observed the dog during three hours

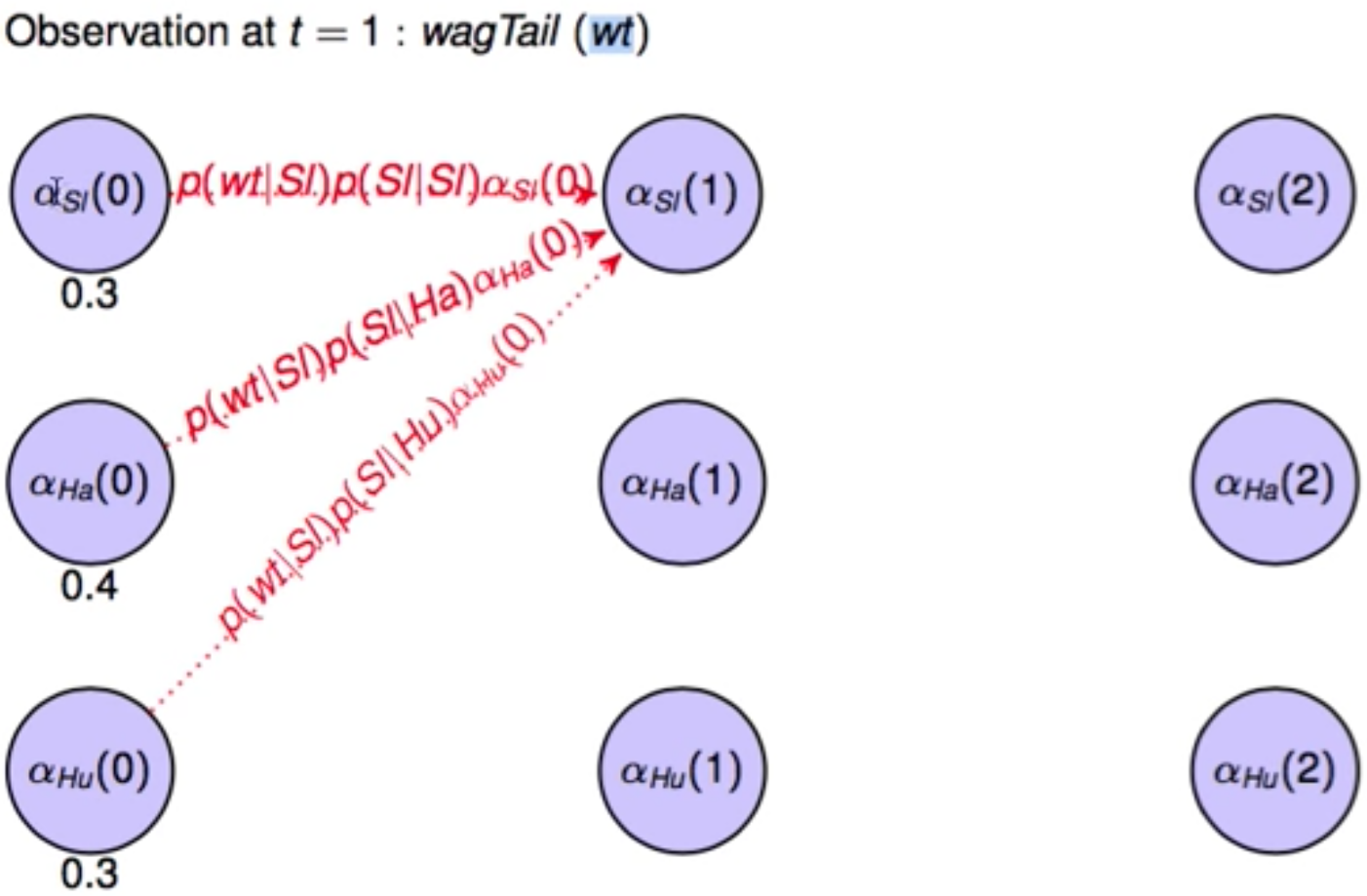
The behaviors at every hour were:
$\{wagTail_1, restless_2, wagTail_3\}$

1. What is the most likely state the dog can be in?
2. What is the probability distribution of the states the dog can be in?

The next slides demonstrate the computations with only two time slices $t \in \{1, 2\}$

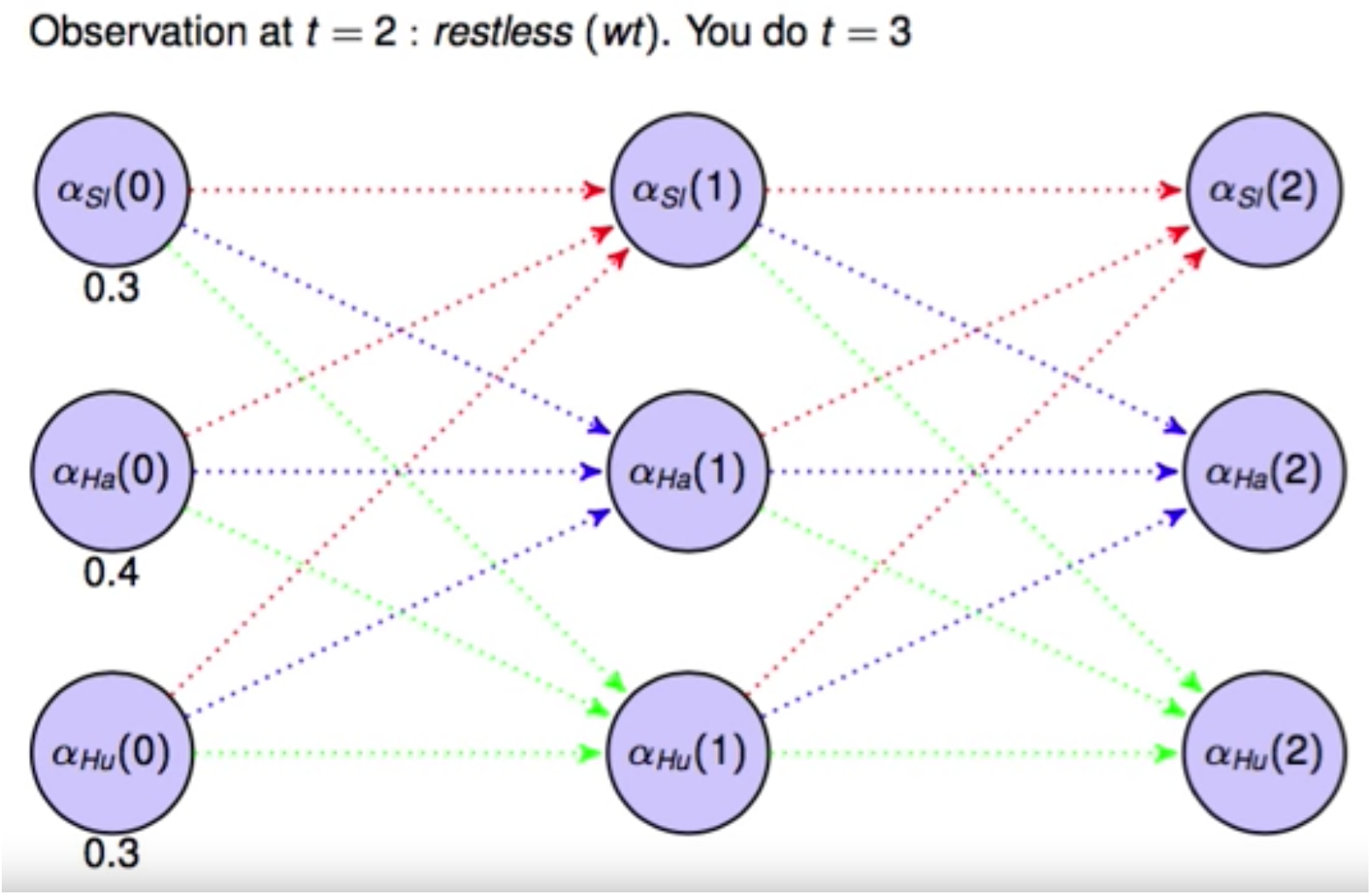
The Forward Algorithm

Trellis for the states of the dog given observations



The Forward Algorithm

Trellis for the states of the dog given observations



The Decoding Problem

Most probable sequence of states given observations

The goal is given some observations, say {*Restless*, *WagTail*, *WagTail*, *Growl*} what is the most likely sequence of states?

Just like the trellis, but maintaining only the highest probablity at each timestep *t*

Viterbi's Algorithm

https://www.youtube.com/watch?v=6JVqutwtzmo&t=178s